SOMMAIRE

INTRODUCTION..... 2

I. Cahier des charges..... 3

1. Cadre du projet..... 3

2. Objectifs du projet..... 3
a. Les véhicules..... 3
b. Les caméras..... 4
c. Environnement du jeu..... 4
d. Gameplay..... 4
e. Animation et collisions..... 4

II. Dossier de conception..... 5

1. Choix de conception..... 5

2. Interface homme – machine..... 6

3. Techniques utilisées..... 7
a. Modélisation sous 3DS Max..... 7
b. Réalisation dans Virtools..... 8

4. Problèmes rencontrés..... 10

III. Guide d'utilisation..... 11

1. Installation – Utilisation..... 11

2. Détails des contrôles..... 12

3. Détails des actions possibles..... 13

IV. Bilan..... 13

CONCLUSION..... 13

INTRODUCTION

Dans le cadre de l'UV VI50, il nous a été demandé de réaliser un jeu vidéo. Celui-ci ayant pour but de mettre en application toutes les notions vues que ce soit en modélisation, animation… à l'aide du logiciel 3DSmax ou en programmation en script à l'aide de Virtools.

La principale difficulté est de trouver un compromis entre système temps réel et jeu à la fois de qualité et fonctionnel.

Nous avons décidé de réaliser un jeu de course. Nous allons tout d'abord vous présenter le cahier des charges que nous avons réalisé puis le dossier de conception et enfin le guide d'utilisation du jeu.

I. Cahier des charges

1. Cadre du projet

Ce projet, réalisé dans le cadre de l'UV VI50 : Vision et réalité virtuelle, doit permettre aux étudiants de mettre en pratique l'ensemble des notions techniques et théoriques d'imagerie 3D et de réalité virtuelle vues dans cette UV ainsi que dans l'ensemble des UV d'imagerie de l'UTBM.

L'ensemble des modèles et de l'univers virtuel sera réalisé à l'aide des logiciels 3D Studio Max 6 et Virtools 3.5.

2. Objectifs du projet

L'objectif à atteindre est de réaliser un jeu vidéo de simulation de course proposant l'ensemble des fonctionnalités classique en la matière :

- Au moins un circuit avec des décors en accord avec l'environnement du jeu.
- Des véhicules et personnages animés.
- Des interactions avec le clavier pour piloter les véhicules.
- Une gestion des collisions
- La possibilité des sélectionner le point de vue parmi plusieurs caméras.
- Un compteur de temps et de tours.
- Un moyen de sauvegarder les meilleurs temps (dans un fichier en créant notre propre "Building Block").
- Une carte du circuit en cours avec la position de son véhicule.

Pour rendre les jeu plus attrayant, nous avons l'intention de mettre en place des éléments supplémentaires comme :

- Un mode de jeu contre une IA.
- La possibilité d'utiliser des bonus.
- Des effets graphiques et sonores avancés.

a. Les véhicules

Le joueur aura le choix entre plusieurs véhicules pour participer à une course :

- Karting
- Moto Cross
- Quad

Les véhicules auront des vitesses maximales en marche avant ainsi qu'en marche arrière. Les accélérations et décélérations seront progressives.

b. Les caméras

Le jeu proposera 4 points de vue différents :

- Une vue de derrière de type 3ième personne.
- Une vue "intérieure".
- Une vue de devant de type 1ière personne.
- Une vue rétro.

c. Environnement du jeu

Le thème du jeu sera axé sur la musique et ses différents styles. La course mettra en scène les Dieux de la musique chacun correspondant à un style de musique différents (Rap, Rock, Disco). La course se déroulera dans un monde paradis / enfer. Des obstacles seront placés à différents endroits du circuit. Pour rester dans l'environnement du jeu, ceux-ci prendront la forme d'objets ou de monstre des enfers (fourche de diable, Cerbère, …).

d. Gameplay

L'élément principal d'un personnage lors d'une course sera une "barre de pouvoir". Celle-ci augmentera au fur et à mesure de la course. Une fois au maximum, le joueur pourra déclencher son "Combo" permettant d'augmenter sa vitesse ou de diminuer celle de ses adversaires.

D'autres bonus seront proposés en plus de ceux augmentant la "barre de pouvoir", un joueur pourra, par exemple :

- Diminuer la "barre de pouvoir" des autres joueurs.
- Lancer un éclair sur un concurrent pour le ralentir.

e. Animation et collisions

Le "Combo" prendra la forme d'une animation mettant en scène le personnage dans une situation caricaturale de son style de musique. Par exemple, le rockeur s'énervera sur sa guitare électrique, …

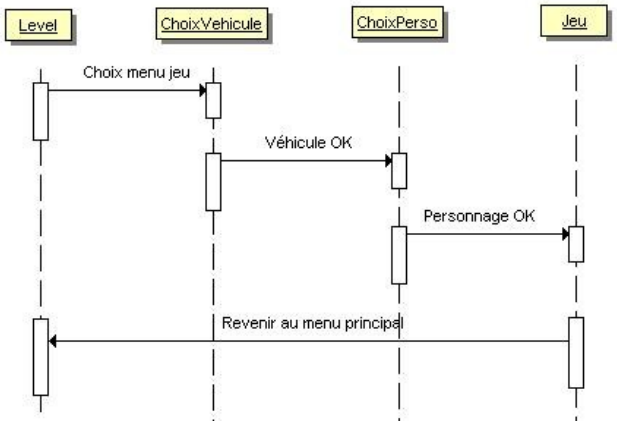
D'autres animations seront mise en place comme les roues des véhicules qui se braquent ou les personnages qui font un geste quand on les choisit.

Pour avoir un comportement "réaliste" au niveau de la gestion des collisions, les véhicules "glisseront" sur les murs en fonction de leur angle de collision.

II. Dossier de conception

1. Choix de conception

Nous avons décidé de mettre en place plusieurs scènes pour mieux gérer notre application. Au final, nous avons trois scènes (choix du véhicule, choix du personnage, jeu) et le level (qui contient les menus principaux). Chacune des scènes s'enchaînant au fil des choix de l'utilisateur. Cela permet également de pouvoir recommencer le jeu.



Au niveau des scripts, nous avons préféré faire des scripts "génériques" contenus dans le level. Cela évite de recopier les mêmes scripts dans les différents objets auxquels ils s'appliquent. Cette façon de procéder implique que les scripts sont présents dans toutes les scènes ce qui nous a obligé à mettre en place un système d'envoi et de réception de messages.

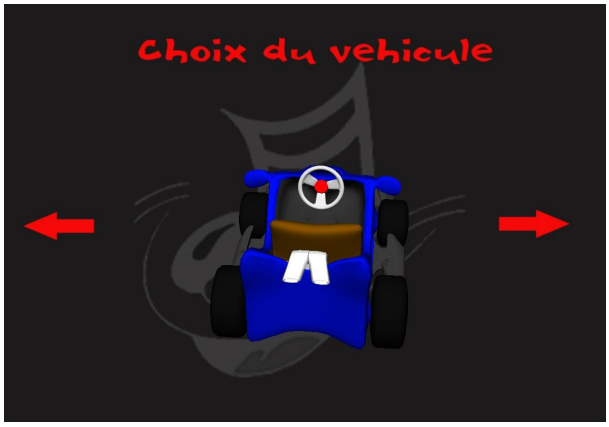
Cependant, certains scripts ont du être impérativement crées dans des objets (principalement des 2D Frames et les characters).

Pour gérer les meilleurs scores, nous avons décidé de nous créer nos propres Building Blocks grâce à Visual Studio. Ceux-ci formeront une DLL qu'il faudra copier dans le répertoire Building Block de Virtools.

2. Interface homme – machine

La navigation dans les menus de l'application ainsi que les déplacements et les actions durant le jeu se fait exclusivement au clavier. Nous avons décidé d'implémenter seulement une gestion du clavier car nous trouvions qu'il était plus facile de se servir d'un périphérique d'entrée plutôt que de passer de l'un à l'autre sans arrêt.

Choix des véhicules



Un écran présente les véhicules individuellement en les faisant tourner sur eux-mêmes. L'utilisateur passe d'un véhicule à l'autre grâce aux touches gauche et droite et valide son choix par la touche entrée.

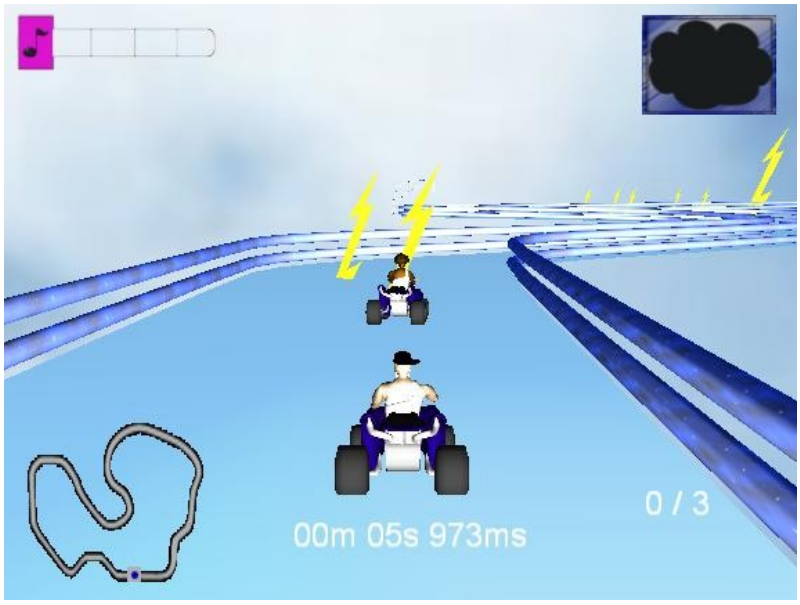
Choix des personnages

L'écran suivant permet de choisir parmi les trois personnages proposés : Un Rockeur, un Rappeur et un Disco.

Les informations utiles au joueur lors de la course sont disponibles dans plusieurs 2D Frames dispersées autour de l'écran. Ainsi l'utilisateur peut connaître, lors de sa course, divers paramètres comme son temps, sa barre de combo, son bonus courant…

Barre de combo

Bonus courant



Mini–map

Compteur de tours

Temps

3. Techniques utilisées

a. Modélisation sous 3DS Max

Modélisation

A la différence du cinéma et des millions de polygones dans les productions cinématographiques, le jeu vidéo fait appel à la puissance d'affichage de notre ordinateur. Dans ce cas, nous avons du penser autrement, afin d'offrir aux joueurs un confort de jeu idéal. Il nous a donc fallu optimiser nos modèles (en terme de nombre de polygones) tout en essayant de garder les détails importants des modèles.

Nous nous sommes donc définit un nombre maximal de vertices pour chacun de nos modèles : 10000 pour les persos comme pour les véhicules.

Tous les modèles ont été réalisés à partir d'une forme de base (le cube), puis retouchés en "polygone éditable".

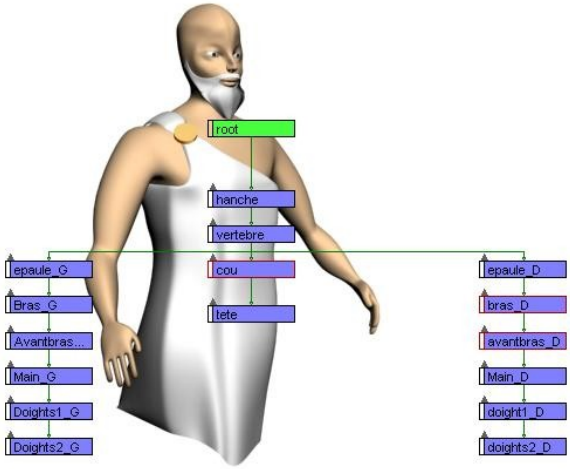
Mapping et textures

Pour un rendu fidèle à nos attentes, les modèles ont été mappés, puis texturés.

Skinning et animation

Si l'animation des roues des véhicules a été réalisée sous Virtools, les animations des personnages ont été réalisés sous 3ds max.

Pour cela, il nous a fallu définir le squelette du personnage grâce aux "bones" de 3ds max.



Le root est la base du squelette. Viens ensuite les hanches puis la colonne vertébrales. Ensuite à cette colonne vertébrale sont attachés les deux membres supérieurs. Ces derniers sont composés d'une épaule, d'un bras, d'un avant bras d'une main, et des deux bones qui permettent de contrôler les doigts. Nous n'avons pas jugé nécessaire de définir des bones pour chaque doigt. Et enfin, le cou et la tête viennent se rattacher à la colonne vertébrale.

Ensuite grâce au modificateur « skin », nous avons défini les zones d'influence de chaque bone sur le corps du personnage. Une fois cette étape cruciale terminée, l'animation des personnages ne consiste plus qu'à déplacer les bones comme voulu.

Cinématiques

Dans l'optique de rendre notre jeu le plus vivant et dynamique possible, nous avons décidé de réaliser des cinématiques. Pour cela, nous avons réalisé des animations sous 3ds max puis importé le rendu de ces dernières dans un logiciel de montage vidéo : Sony Vegas.

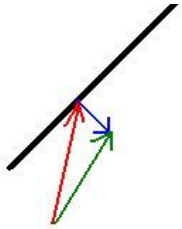
Ce logiciel n'étant pas gratuit nous avons utilisé une version d'évaluation disponible sur le site officiel.

b. Réalisation dans Virtools

Gestion des collisions

La gestion de collision sous virtools se fait à l'aide du Building Block "Collision Detection". Celui-ci retourne toutes les collisions entre un objet voulu et l'ensemble des objets de la scène ayant un attribut "collision manager".

Dès qu'une collision est détectée, le Building Block nous renvoie, entre autre, l'objet ainsi que le numéro de sa face en collision. Cela nous permet de récupérer la normale à cette face. Il est également facile de récupérer le vecteur direction du véhicule.



En faisant une rotation de l'angle entre le vecteur rouge (vecteur direction du véhicule) et le vecteur vert (vecteur somme du vecteur direction du véhicule et de la normale à la face) on simule un effet de glissement le long du mur en le rasant.

Cependant, dans certains cas, il est possible qu'après la rotation, le véhicule soit toujours en collision. Dans ce cas, il est nécessaire de faire une légère translation du véhicule vers l'intérieur de la route.

Intelligence Artificielle

Nous avons implémenté une intelligence artificielle en utilisant le Building block « Grid path solver ». Nous avons choisi cette méthode afin d'utiliser les notions vues en cours de recherche de plus court chemin. Pour cela, nous avons tout d'abord crée une grille et placé un calque de collision représentant les murs du circuit. Nous avons ensuite paramétré l'IA en utilisant la méthode de la distance Euclidienne. Le Grid path solver va rechercher le plus court chemin entre la position de départ de l'IA et le premier checkpoint. Une fois arrivé à ce point, un nouveau chemin entre le checkpoint 1 et 2 est recherché et ainsi de suite. La vitesse de déplacement et la direction sont paramétrés par une « Linear progression » afin d'obtenir un déplacement fluide. Le temps imparti pour chaque itération est paramétré en fonction de la distance à parcourir avant le point de passage suivant (pour obtenir une vitesse constante) et un multiplicateur permettant d'ajuster la vitesse de l'IA.

Gestion des cameras

Les différentes caméras sont placées grâce à leur propre script. Chaque script contient les Building Block "Keep at constant distance" et "Look at". Cela permet à la caméra de suivre le véhicule tout en le regardant.

Ensuite, un script permet de définir la caméra active lorsque l'on appuie sur les touches de changement de vues.

Classement

Pour calculer la position relative du joueur par rapport à l'IA, on utilise une curve centrée sur le circuit. On calcule le point projeté des véhicules sur la curve, et on en déduit la distance qu'il leur reste à parcourir jusqu'à la ligne de départ. En croisant la position relative dans le tour au nombre de tours déjà effectués par les deux véhicules, on peut déterminer leurs positions relatives.

Gestion des meilleurs scores

Les meilleurs scores dans le jeu sont stockés dans un fichier .txt sous la forme : Pseudo – Score. Pour gérer les meilleurs scores dans le jeu, nous avons crée trois Building Blocks :

- EstMeilleurScore qui prend un temps en paramètre et qui renvoi VRAI si le temps fait partie des meilleurs.
- WriteScore qui attend un pseudo et un score pour les écrire dans le fichier.
- GetMeilleurScore qui retourne les trois meilleurs couples Pseudo – Score.

Déplacement du véhicule

Les déplacements du véhicule sont gérés par trois boucles « while » contrôlées par des Building Block "Key Event". Tant que la touche « haut » est activée, une boucle augmente la vitesse de déplacement (jusqu'à une vitesse maximale) puis translate le véhicule de ce vecteur vitesse. On réalise le même traitement pour la touche bas. Lorsqu' aucune de ces touches n'est activée, on ramène la vitesse progressivement vers 0 pour que le véhicule s'arrête peu à peu.

Pour prendre un virage, il suffit d'effectuer une rotation autour de Y. Lorsque le véhicule tourne, ses roues tournent également (ou la moto se penche). Le rayon de braquage des roues est limité à un certain angle, et les roues se reviennent en position droite automatiquement dès que l'on arrête de tourner.

Bonus

Les bonus sont séparés en deux scripts, le premier permet d'obtenir un bonus lorsque qu'une collision entre le véhicule et un des bonus a lieu. Le second permet la gestion de ces bonus, le plus et le moins ont un effet immédiat sur la barre de combo. L'éclair quant à lui a une gestion complexe. En effet nous avons tout d'abord vérifié si celui-ci entrait en collision avec l'IA, ou bien avec les barrières du circuit. Ensuite nous avons veillé à ce que lorsque l'éclair vient à disparaître suite à l'une des conditions ci-dessus, il se place à nouveau dans la même position et orientation que le véhicule.

Combo

Dès que le combo est lancé, nous lançons alors une vidéo représentant le combo du personnage en mettant auparavant en pause tous les autres éléments comme le chronomètre et l'IA. Une fois la vidéo terminée, nous remettons en marche les éléments en pause et appliquons les effets du combo, à savoir le boost pour une durée déterminée.

Minicarte

Pour réaliser la minicarte, nous avons utilisé deux 2Dframes que l'on déplace l'une par rapport à l'autre. On calcul la position du véhicule par rapport au circuit et, après mise a l'échelle de la minicarte, on place le point à l'endroit correspondant sur la minicarte.

4. Problèmes rencontrés

Création de la DLL

Pour créer nos Building Block de gestion des meilleurs scores, il nous a fallu comprendre les mécanismes de fonctionnement de ce type de DLL. Une fois le squelette créé, il nous a été possible de coder le comportement des Building Block. Une des difficultés a été de récupérer et de manipuler des données de type String compatible entre Visual Studio et Virtools.

Gestion des collisions

Plusieurs problèmes sont survenus lors de la création du script de gestion de collisions. Les collisions n'étaient pas toutes bien détectées : il nous a fallu mettre la détection en type face et appliquer l'option "set as unit" sur les véhicules.

Ensuite les valeurs retournées par le Builiding Block "Detection Collision" ne sont pas forcement celle que l'on attendaient. Par exemple, les valeurs "impact normal" et "impact orientation" sont calculées à partir des deux points les plus proche dans les deux objets en collision. Par contre, un des paramètres de sortie est le numéro de la face en collision et le nom de l'objet en collision. Nous avons donc du, grâce à des Building Block "Op", récupérer le mesh puis la normale à la face en collision. Ensuite, il nous a fallu récupérer le vecteur direction du véhicule. Pour cela, il a fallu comprendre ce qu'étaient les vecteurs "get right" et "get dir".

Communication 3DS Max / Virtools

Après avoir modélisé quelquechose sous 3DS Max, il faut l'exporter au format .nmo pour l'utiliser dans Virtools. Cependant, plusieurs problèmes peuvent intervenir durant cette phase d'importation / exportation, principalement au niveau des matériaux et textures.

III. Guide d'utilisation

1. Installation – Utilisation

Le jeu est lancé dans un navigateur web via le plugin virtools 3D Life Player qui peut être téléchargé à l'adresse suivante.

Une fois dans le jeu, divers menus apparaissent. Le premier permet de choisir de jouer directement ou de consulter l'aide.

Si on décide de consulter l'aide, il suffit d'appuyer sur ECHAP afin de revenir au menu.

Lorsque l'on décide de jouer, on arrive à la sélection du véhicule, pour faire défiler ceux-ci, il suffit d'appuyer sur les touches Gauche et Droite. Une fois le véhicule validé en appuyant sur la touche ENTREE, on choisit alors son personnage suivant le même principe.

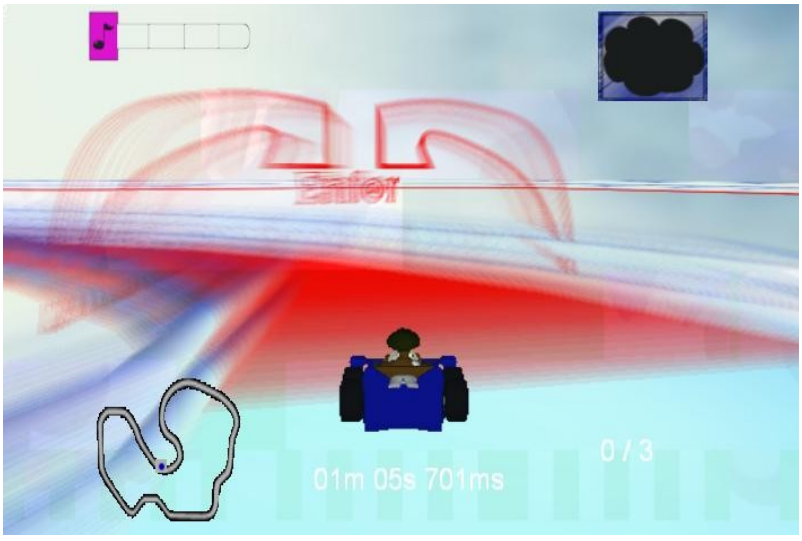
Lors du jeu, l'utilisateur doit réaliser trois tours de circuit tout en étant premier à la fin des trois tours. Si à la fin de la course ; le joueur réussit à battre un highscore, il aura alors la possibilité d'enregistrer son nouveau score.

Dans tous les cas, à la fin du jeu, l'utilisateur peut voir la liste de tous les highscores et peut alors retourner au menu principal en utilisant la touche ENTREE.

Lors du jeu, l'utilisateur se déplace sur le circuit en utilisant les touches fléchées. Divers bonus sont présents sur le circuit :

- Plus : Permet d'augmenter d'un cran sa barre de combo. Ce bonus a un effet immédiat.
- Moins : Diminue la barre de combo d'un cran. Ce bonus a un effet immédiat.
- Eclair : Permet de lancer un éclair en utilisant la touche ESPACE. Si l'éclair vient à toucher l'adversaire, celui-ci sera ralenti.

Le joueur possède une barre de combo qu'il pourra utiliser comme bon lui semble. Une fois la barre entièrement remplie, il peut alors appuyer sur la touche C et déclencher alors son combo spécial. Suite à celui-ci, la vitesse du joueur augmentera pendant un certain temps avec un effet de motion-blur (image floutée).



2. Détails des contrôles

Les contrôles se font exclusivement au clavier. Voici les touches utilisées :

- Flèches : permettent de diriger le véhicule
- C : déclenche le combo
- Espace : tirer un éclair
- Page suivante/précedente : permet de changer de caméra
- Ctrl droit : vue rétroviseur

Tous ces contrôles sont récapitulés dans une page d'aide accessible depuis le menu principal du jeu.

3. Détails des actions possibles

Chacun des véhicules peut se déplacer librement à l'intérieur du circuit (marche avant, marche arrière, demi-tour). Si on touche un des bords du circuit, le véhicule réagit automatiquement en glissant le long de celui-ci tout en s'en rapprochant.

Tout au long du circuit est disposé un ensemble de bonus qu'il est possible de ramasser en passant dessus. Certains bonus ont un effet immédiat (augmentation et diminution de la barre de combo), d'autres se lancent quand le joueur le décide (éclair). Une fois que la barre de combo est remplie, on déclenche le combo quand on le souhaite. A partir de ce moment une vidéo mettant en scène le personnage démarre. Lorsque le jeu redémarre, l'effet du combo commence et le véhicule roule plus vite pendant quelques secondes.

Pendant la course, le joueur peut changer librement de vues parmi les quatre proposées (Vue 1ère personne, Vue 3ème personne, Vue intérieure et Vue rétroviseur).

A la fin de la course, si le score réalisé fait partie des trois meilleurs, le joueur devra saisir son pseudo pour enregistrer son score. Une fois cela réalisé, il est possible de revenir au menu principal pour recommencer la course avec un autre personnage et un autre véhicule.

IV. Bilan

Malheureusement, nous n'avons pu terminer ce projet comme nous le souhaitions et certains points auraient pu être améliorés. Quelques mois ne suffisent pas pour réaliser un projet d'une telle ampleur et de nombreuses fonctionnalités liées à un jeu de course comme plusieurs concurrents n'ont pu être implémentées.

Même si nos ambitions au début de ce projet étaient démesurées, nous sommes tout de même satisfait du résultat final car nous avons réussi à proposer un jeu globalement fonctionnel.

CONCLUSION

Ce projet nous a permis de comprendre les difficultés pour mettre en œuvre un jeu vidéo, que ce soit au niveau du cahier des charges, au niveau des divers éléments à créer ou encore au niveau de la programmation.

Il nous a également permis de mettre en application toutes les notions vues au cours de l'UV VI50 dans divers domaines (modélisation, mapping, animation, rédaction de script, utilisation de VSL…).