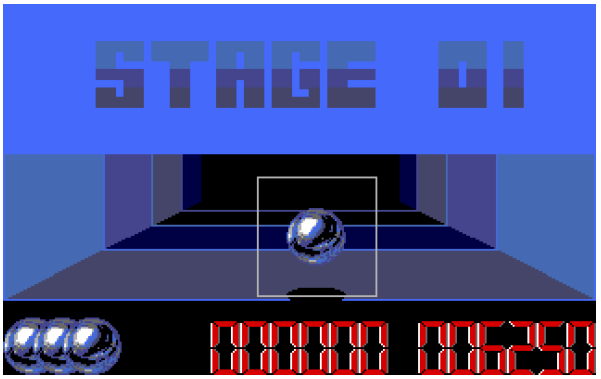
Synthèse d'images I
Projet : The IMAC Light Corridor

Il est temps de mettre en pratique les nouvelles notions obtenues pendant les cours de Synthèse d'Images. Pour se faire, vous devrez concevoir un jeu vidéo relativement simple, inspiré plus que largement sur un (très) ancien jeu vidéo appelé "The Light Corridor". Pour avoir une bonne vision de ce qui est attendu, et pour vous donner des idées, vous trouverez un walkthrought complet de ce jeu Amiga, Atari ST...



Vous allez donc devoir concevoir votre version de ce jeu vidéo en `C/C++` avec OpenGL. Le but étant :

- De vous familiariser avec la programmation de projet un peu plus complexe.
- De vous faire utiliser les fonctions de dessin 3D OpenGL et appréhender les problématiques de construction 3D
- De vous faire implémenter une application interactive fonctionnelle.

Modalités de rendu

- **Nombre de personnes par projet :** 2 ou 3 (du travail supplémentaire pour les trinomes est requis)
- **Livrable :**
  — Les sources C/C++ ainsi qu'un système de compilation permettant de compiler et d'exécuter notamment sur l'environnement linux des machines de l'université. Pour ce faire, un système de type Git sera mis en oeuvre et mis à ma disposition.
  — Un court rapport au format PDF décrivant votre projet : son mode d'installation (compilation et exécution), son fonctionnement (commandes utilisateurs), les résultats obtenus (fonctionnalité implémentées, capture d'écran, timing...), votre méthode de travail (très brièvement), des détails techniques si vous souhaitez détailler une fonctionnalité ou un algorithme jugé original, les difficultés rencontrées et enfin les améliorations possibles.
- **Soutenance :** Si les conditions le permettent, vous pourrez défendre votre projet durant une soutenance en faisant une démonstration de votre projet et en reprenant les différents points du rapport. Votre présentation durera au maximum 5 minutes (c'est court) et sera suivi d'éventuelles questions du jury pour compléter votre soutenance.
- **Date de rendu :** *non défini*
- **Date de soutenance :** *non défini*

1 Présentation générale du jeu

Le joueur dispoe d'une raquette carrée qui évolue en ligne droite dans un corridor présentant bonus et obstacles. La raquette permet de lancer et de faire rebondir une balle le long de ce corridor. La raquette avance le long de ce corridor mais reste toujours devant la balle (il ne peut pas la dépasser).

Lorsque la balle rebondit sur les murs ou les obstacles, elle peut revenir vers le joueur (la raquette) et si elle dépasse la position de la raquette, le joueur perd une vie. Le but est d'aller le plus loin le long du ... IMAC light corridor !

2 Eléments du jeu

2.1 Le corridor

Le corridor est un long tunnel rectangulaire. Il est constitué de sections qui peuvent contenir un ou plusieurs obstacles (essentiellement des murs) et des bonus (des objets modifiants la raquette, la balle, voire le corridor lui-même).

Eventuellement (cf. section 4), ces sections sont regroupées en niveau qui permettent de re-

partir, en cas de perte d'une vie, au début du niveau, au lieu du début du jeu.

2.2 La raquette

Le joueur est symbolisé par sa raquette carrée (ou rectangulaire au choix). Celle-ci peut éventuellement changer de taille suivant les bonus récupérés. La raquette reste dans "l'embrasure" (section rectangulaire) du corridor, et elle ne peut pas en sortir, elle est donc arrêtée par les murs du corridor. Fort logiquement, la raquette ne peut pas traverser les obstacles le long du corridor (cf. section 3.1).

La raquette dispose d'un nombre de vie (initialement 5). Si la balle "dépasse" la raquette, vers la caméra, alors le joueur perd une vie. Si le nombre de vie est égal à zéro, la partie est perdue. Sinon, à la perte d'une vie, la balle revient au centre de la raquette et le joueur peut positionner, sans pouvoir avancer, sa raquette dans le corridor avant de la renvoyer.

2.3 La balle

La balle est sphérique. Elle rebondit sur les bords du corridor, les obstacles, et sur la raquette (voir section 3.2). Tout rebond sur les bords du corridor et les obstacles se fait comme au billard, c'est à dire dans la direction symétrique par rapport à la normale.

2.4 Les obstacles

Ce sont de simple panneaux rectangulaires qui sont dans les sections du corridor. Ils sont perpendiculaire à l'axe principal de celui-ci (la direction du corridor). La balle rebondit dessus. Leur forme peut éventuellement différer d'un rectangle mais vous devrez, dans ce cas, prendre en considération l'interaction avec la raquette (cf. section 3.1)

2.5 Les bonus

Ce sont des objets 3D que vous modéliserez à votre souhait. La balle les traverse. Quand la raquette traverse le bonus, il est pris par le joueur.

Vous devrez au minimum créer deux bonus. Le premier est une colle qui permet de garder la balle lorsqu'elle revient sur la raquette. Elle peut être renvoyé au clic droit. Le second permet de gagner une vie.

3 Interface Homme-Machine et moteur de jeu

Dans cette section, vous trouverez les différentes possibilités du joueur ainsi que des éléments du moteur de jeu de cette aplication interactive 3D.

3.1 Mouvement de la raquette

Au départ du jeu, et à chaque nouvelle vie, la balle est attachée au centre de la raquette. Lorsque la souris se déplace sur l'écran de jeu, la raquette bouge en regard c'est à dire que le centre de la raquette suit le curseur de la souris, sans toutefois pouvoir dépasser les bords du corridor. Au clic droit la balle est envoyé de la raquette en ligne droite devant elle le long du corridor.

La raquette est déplacable à la souris. Au clic gauche de la souris, la raquette avance si elle le peut. En effet, la raquette doit être arrêtée par les obstacles. Cela veut dire que si la raquette recouvre une partie d'un obstacle, elle ne peut plus avancer. Vous pouvez faire ce "test" légèrement en amont de l'obstacle lui-même.

3.2 Les rebonds

Comme indiqué précédemment la balle rebondit sur les murs et sur les obstacles symétriquement par rapport à la normales du mur impactée. Les équations dans ce cas sont assez simple car toutes les surfaces de rebond sont parallèles aux axes.

Pour la raquette la balle rebondit dans une direction fonction de la distance du point d'impact de la balle au centre de la raquette. Nous vous laissons libre de choisir la formule de la direction de ce rebond (qui devra être indiqué dans le rapport).

3.3 Menu de départ et de fin

Votre application devra proposer un menu de départ permettant à minima de lancer le jeu ou de quitter le jeu. En cas de victoire (arrivée à la fin du corridor) ou d'échec (plus de vie), le jeu présentera une page dédiée.

3.4 Illumination

Une source de lumière ponctuelle devra être placée au niveau de la caméra. Par ailleurs, vous positionnerez une autre source de lumière au niveau de la sphère. De plus, au moins un élément de votre jeu devra être texturé (balle, décors, obstacle, raquette).

4 Travail supplémentaire pour les trinômes

Voici les éléments supplémentaires à coder pour les trinômes :
- Gestion de niveau. Le corridor est segmenté en niveau. Chaque niveau est chargé **indépendament** au fur et à mesure de la progression du joueur. Un code existe et permet de commencer directement à ce niveau.
- Le menu de départ permet de commencer par un niveau ou de (re)commencer au départ
- Un score est défini et est affiché en direct dans le jeu. Ce score s'incrémente suivant la distance parcourrue, les bonus ramassés, etc. Le nombre de vie et les éventuels bonus sont également indiqués visuellement dans le jeu.
- L'affichage de fin est un menu qui permet de revenir au début du jeu et affiche le score.
- Au moins un élément de votre jeu sera transparent.

5 Remarques et Conseils

Tout d'abord, voici un aperçu des éléments les plus complexes à prendre en compte.
- La gestion du rebond, plus particulièrement celui de la balle sur la raquette (les autres sont simples)
- La gestion de la modélisation du corridor et de son parcours. Vous êtes libre sur ce point mais certaines modélisations permettent de se simplifier la vie, d'autres sont plus efficaces.
- La gestion des collisions raquette-obstacle qui peut être complexe.

Quelques conseils enfin :
- Il est très important que vous réfléchissiez avant de commencer à coder aux principaux modules, algorithmes et aux principales structures de données que vous utiliserez pour votre application . Il faut également que vous vous répartissiez le travail et que vous déterminiez les tâches à réaliser en priorité.
- Ne rédigez pas le rapport à la dernière minute sinon il sera bâclé et cela se sent toujours.
- Le projet est à faire en groupe. Il est impératif que chacun d'entre vous travaille sur une partie et non pas tous "en même temps" (plusieurs qui regarde un travailler). sinon vous n'aurez pas le temps de tout faire. De toutes façons le Git sera consulté pour vérifier que chaque membre du groupe a bien contribué.
- N'oubliez pas de tester votre application à chaque spécification implémentée. Il est impensable de tout coder puis de tout vérifier après. Pour les tests, confectionnez vous tout d'abord de petites scène. Si cela fonctionne, vous pouvez passer à plus gros ou plus complexe.
- Vos chargés de TD et CM sont là pour vous aider. Si vous ne comprenez pas un algorithme ou avez des difficultés sur un point, n'attendez pas la soutenance pour nous en parler.

6 Pour aller plus loin

Vous pouvez et nous vous encourageons à vous rapprocher un peu plus du jeu d'origine en terme de complexité ou de personnaliser le jeu pour y mettre votre empreinte parce que c'est plus fun :p et c'est un plus pour la notation.

Notamment vous pouvez :

- créer des bonus différents.
- créer des boss de "fin de niveau". Chaque boss doit être touché par la raquette un certain nombre de fois pour passer.
- créer des obstacles complexes.
- donner la capacité à faire revenir en arrière la raquette.
- ...