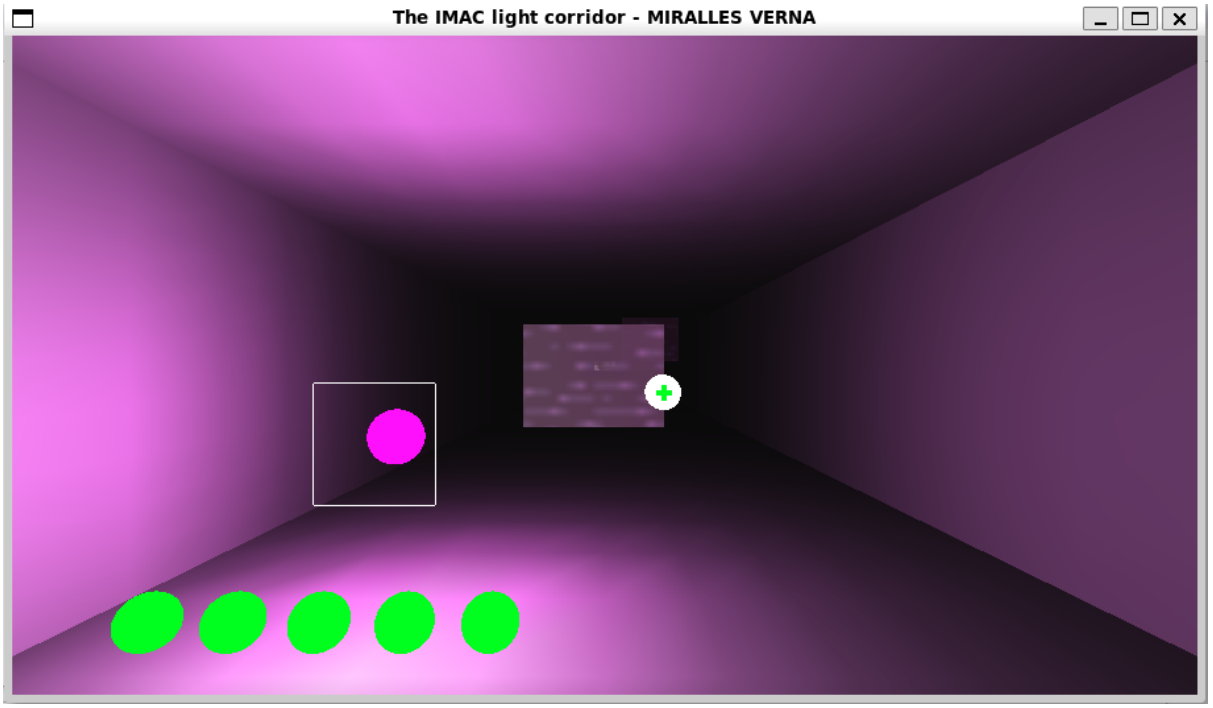
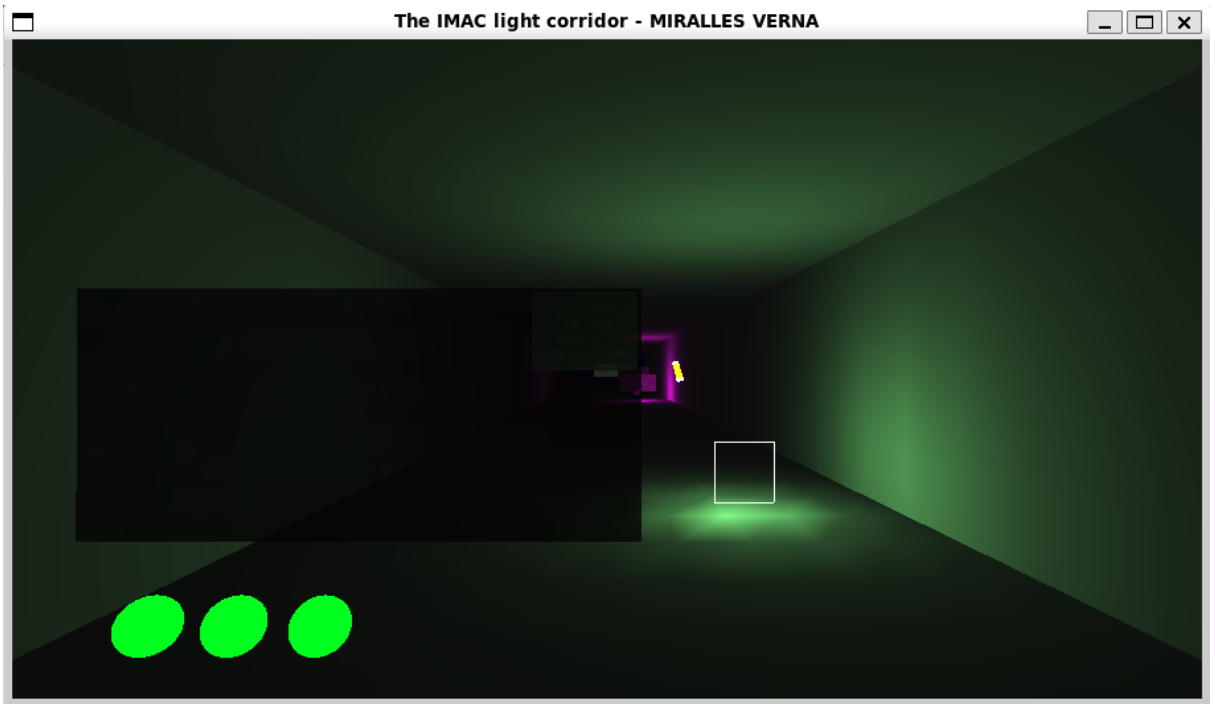
Rapport de projet





Mode d'installation :

Notre projet se compile avec Make ("make projet" sous linux) et s'exécute en appelant la commande ./bin/projet.out.

NOTE IMPORTANTE : dans les constantes définies en en-tête du fichier projet.c, on trouve SPEED et RACKET_SPEED. Ces valeurs peuvent-être à modifier pour que l'expérience jeu soit agréable. En effet, la vitesse varie selon les ordinateurs sur lesquels le programme est lancé… Plus d'explications dans la partie expliquant nos difficultés.

Fonctionnement :

Liste des commandes utilisateur :

Clic gauche : la balle est envoyée en ligne droite lorsqu'elle est collée à la raquette au début de partie, après la perte d'une vie, ou après avoir récupéré un tube de colle.

Clic droit : la raquette avance dans le couloir si elle n'est pas bloquée par un obstacle qui serait devant elle.

Espace : le jeu est mis en pause, et sort de son état de pause après un nouvel appui sur espace.

L : permet au joueur de jouer en mode wireframe plutôt que shading plein

P : permet au joueur de jouer en mode shading plein plutôt que wireframe

Q : permet de fermer la fenêtre sans menu

Mouvement de la raquette : notre interprétation du sujet nous a conduit à faire avancer les sections et les obstacles automatiquement, vers la caméra. Sans action de la part du joueur, la raquette se déplace avec le décor, donc en direction de la caméra. Si le joueur clique sur le bouton droit de la souris en revanche, alors la raquette avance dans le couloir.

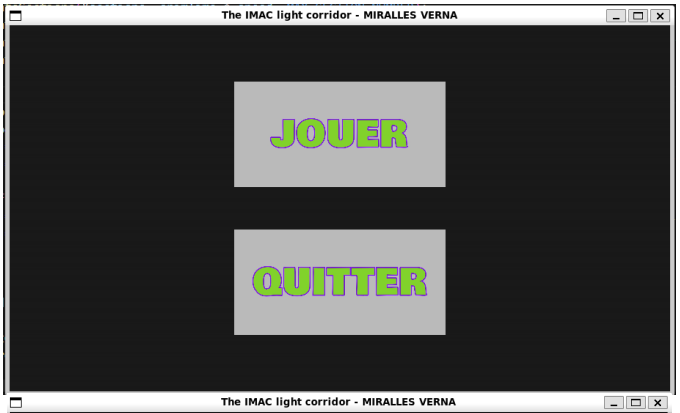
Mouvement de la caméra : plutôt que de faire bouger la caméra, nous avons préféré en donner l'illusion en faisant en sorte qu'elle reste fixe pendant que tout bouge autour d'elle.

Lumière : nous avons décidé de placer la source de lumière au centre de la raquette plutôt qu'au niveau de la caméra, rendant le jeu plus lisible.

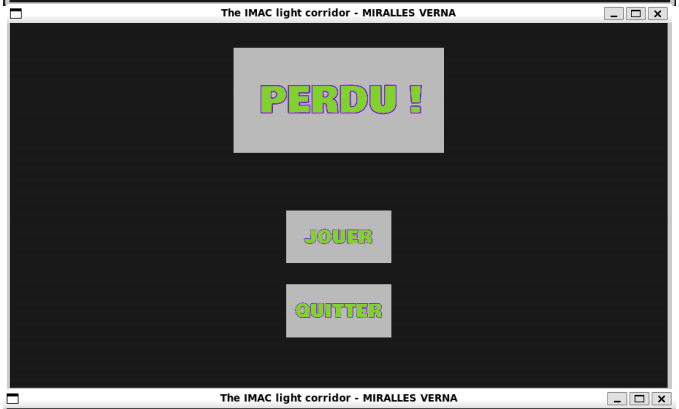
Résultats Obtenus :

Nous avons implémenté 100% des fonctionnalités liées au corridor, la raquette, la balle, les obstacles, les bonus, les menus, et l'illumination. Cependant, quelques petites différences : ce n'est pas la raquette qui porte les vies dans sa structure, mais la balle. Lorsqu'un bonus colle est récupéré, la balle se téléporte directement sur la raquette sans devoir attendre qu'elle ne revienne dessus.
Implémentations supplémentaires : la page de fin est un menu cliquable permettant de rejouer, et les obstacles sont semi-transparents. De plus, le nombre de vies restantes est affiché à l'écran.

Captures d'écran :

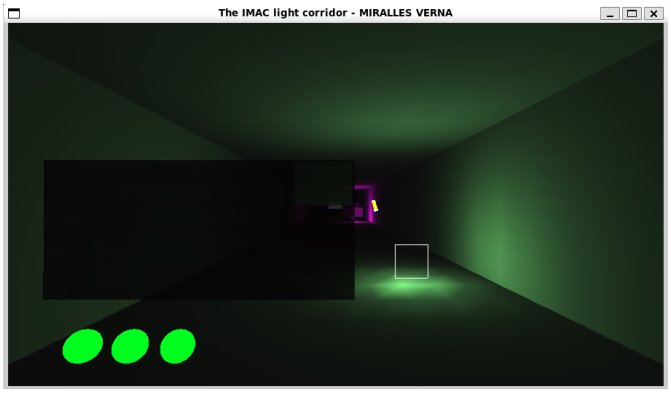


Menu de départ

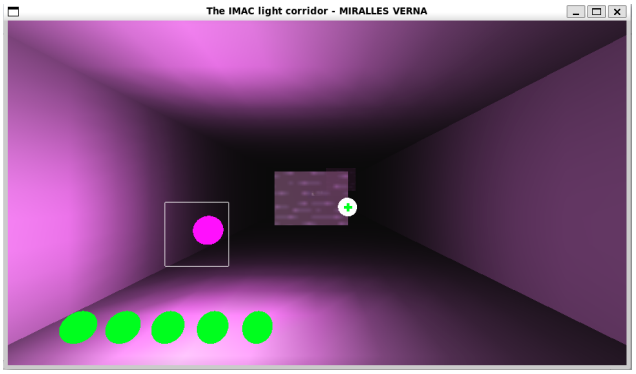


Menu de fin - défaite

Menu de fin - victoire



En bas à gauche : le nombre de vies restantes. Juste au-dessus, un obstacle semi-transparent. A droite, la raquette, avancée dans le couloir. Au-dessus de la raquette, un bonus de tube de colle. Au fond du couloir, la balle émettant une lueur magenta.



La balle collée sur la raquette. A sa droite, un obstacle, texturé et semi-transparent, et un bonus de type vie.

Méthode de travail :

Grâce à github, nous avons pu collaborer, même si nous avons beaucoup travaillé sur le même ordinateur surtout au début du projet. Nous avons travaillé par étapes : d'abord la construction du corridor, puis l'ajout de la raquette, l'ajout de la balle avec ses collisions, puis les obstacles ainsi que leurs collisions et les bonus. Petit à petit, le projet devenait de plus en plus complexe. Pour la répartition du travail, Maxime s'est majoritairement occupé de la physique du jeu (collisions, déplacements, dessins) et Alexandre des interactions utilisateurs (mouvement du curseur, menus) ainsi que de l'organisation et la propreté du code.

Détails techniques :

Maintenant, voyons quelques détails techniques là où ils s'imposent.

Pour le lighting, nous avons choisi d'utiliser les lampes GL_LIGHT directement intégrées à OpenGL. Cependant, sur des faces trop grandes (donc avec des sommets trop éloignés), il était dur d'avoir une illumination correcte. Deux choix s'offraient alors à nous : implémenter des shaders, ou bien subdiviser les faces. C'est ce second choix que nous avons choisi. En effet, les murs étant rectangulaires, il était facile de les subdiviser en les dessinant à l'aide d'une boucle.

Pour le corridor et les obstacles, il s'agit en fait d'un tableau d'une dizaine d'éléments se déplacent tous ensemble en file indienne. Lorsque le premier élément a dépassé la caméra, il est téléporté en fin de file, pour donner l'impression d'un couloir infini. De plus, les obstacles se voient ré-assigner une position et une taille aléatoire. Les obstacles et les sections ne sont téléportés que tant que le nombre maximum de sections dépassées n'est pas atteint ; c'est ce nombre qui définit la distance à parcourir pour gagner (MAX_SECTION_NUMBER).

Le panneau indiquant la distance de fin du jeu est initialisé dès le départ, à une distance correspondant à MAX_SECTION_NUMBER * longueur d'une section. Dès que la raquette traverse ce panneau, la partie est gagnée.

Pour les bonus, nous avons choisi d'en avoir un seul présent simultanément sur le niveau. Ainsi, un seul bonus est créé, et dès qu'il est ramassé ou qu'il passe derrière la caméra, il est téléporté plus loin avec un type et une position aléatoires.

Alternation menus / jeu : tout passe par une variable du nom de "focus" qui indique constamment l'état dans lequel se trouve la partie, et déclenche les fonctions correspondantes, en réinitialisant les acteurs de jeu au besoin, etc.

Difficultés :

La principale difficulté a été le temps imparti pour la création du projet : non pas qu'il ait été trop court, mais couplé aux autres projets il a été challengeant de finir à temps. Mais c'est motivant de travailler, et avec un binôme s'entendant bien, ça le fait !

Une autre difficulté était la gestion de la vitesse. En effet, par défaut, on appelle une fonction permettant de limiter la vitesse à laquelle le jeu se joue, en comparant le temps écoulé entre deux tours de boucle de jeu et le framerate voulu. Cependant dans notre cas comme pour beaucoup d'autres, la vitesse de jeu augmentait dès que le joueur bougeait la souris, et ralentissait quand le joueur ne touchait plus à rien. Nous avons donc décidé de supprimer la ligne limitant la vitesse du jeu ; ainsi il se joue à vitesse constante. Cependant, un autre problème apparaît : le jeu se joue maintenant avec une vitesse proportionnelle à la puissance de calcul de l'ordinateur.. D'où la nécessité d'adapter la vitesse sur chaque nouvel ordinateur.

Possibilités d'amélioration :

Voici une liste des améliorations possibles de ce projet :

Implémenter des fragment shaders, pour pouvoir éclairer correctement des faces peu importe leur taille ; cela éviterait d'avoir à les subdiviser, et donc cela améliorerait les performances du jeu. De plus, il serait plus aisé de les texturer. Bien qu'ayant rapidement utilisé des shaders en cours, les tutoriels disponibles sur internet ne sont pas directement très clairs.

Ajouter des bonus supplémentaires pouvant modifier la raquette, les niveaux

Ajouter des obstacles destructibles à la manière d'un casse-briques

Mieux étudier la référence originale du jeu, pour coller au plus près au sujet.

Compteur de score pour pimenter les parties