**Rapport de projet**

420-SCD-MA Intégration des apprentissages en sciences informatiques et mathématiques

Présenté à

Caroline Houle

Produit par

Olivier St-Jean

Alexandre Hua

Équipe 05, groupe 01

Collège de Maisonneuve

Lundi 4 mai 2015

Table des matières

[1. Introduction 2](#_Toc418523775)

[2. Évolution du projet 2](#_Toc418523776)

[3. Répartition finale des tâches 4](#_Toc418523777)

[4. Complexité du projet 5](#_Toc418523778)

[5. Algorithmes externes 6](#_Toc418523779)

[6. Conclusion 6](#_Toc418523780)

# Introduction

Ce document a pour but de faire le point sur notre document intégrateur en exposant les principales différences entre le produit final et notre soumission initiale, la distribution officielle des tâches, la complexité du projet et les sources externes utilisées.

# Évolution du projet

L'évolution de notre projet était assez atypique par rapport à nos anciennes expériences de développement. À la version pré-alpha, il n'y avait en fait presqu'aucune fonctionnalité disponible. Il s'agissait plutôt d'un squelette pour la suite du développement. Dans notre version alpha, un mode de jeu un peu dénudé a été implémenté et les divers objets de jeu tels que la balle, la batte et les plateformes au sol ont été créées. Ces objets étaient finalement capable d'interagir les uns avec les autres sous les principes de collision. La version bêta était axée sur l'interaction avec l'utilisateur. Plusieurs paramètres, comme la vitesse de la balle, sa masse, sa charge et son comportement ondulatoire, sont disponibles pour être modifiés par l'utilisateur dans le mode libre. L'affichage des informations scientifiques a aussi fait son apparition. Dans la version finale, quelques petits obstacles ont été ajoutés, le mode compétitif (normal) a été complété et la détection des zones pour le pointage est complète. Il y a certaines modifications par rapport à notre soumission:

* La possibilité de mettre le jeu en plein écran ou dans une résolution différente a été retirée. Puisque frapper la balle représente un certain défi, il est plus juste pour ceux jouant à plus grande résolution d'avoir une zone de contact de même dimension relativement à la taille de la zone de jeu. Puisque les résolutions peuvent varier de façon non standard (les formats standards correspondant à 480p, 720p et 1080p), il faut obtenir un ratio personnalisé au lancement de l'application et l'appliquer à tous les éléments, à leurs propriétés et non à la méthode de dessin. Cela aurait nécessité trop de temps considérant les éléments plus importants qu'il fallait ajouter.
* Il était prévu d'afficher le type de la prochaine balle. Cet élément a été retiré car cela posait un problème avec les comportements de la librairie Swing. De plus, nous avons décidé de laisser l'utilisateur choisir un ensemble de balle dans le mode libre. Un type est alors choisi aléatoirement parmi les l'ensemble. Dans le mode normal, ou compétitif, l'utilisateur n'a aucune autorité sur le type de balle qui peut être généré. Afficher la balle dans une étiquette gâcherait le suspense.
* Dans les maquettes de la soumission, les informations scientifiques devaient être affichées dans les deux modes de jeu. Cependant, nous avons ajouté le panneau d'information uniquement dans le mode libre, car il s'agit du mode d'expérimentation. Le mode compétitif ne semblait pas être le mode approprié pour remplir l'interface d'informations. L'affichage des résultats à la fin de la partie a aussi été retirée.
* La possibilité de déplacer la caméra a été ajoutée. Cela nous permet d'observer la trajectoire de la balle, représentée par une série de points. Nous pouvons donc mieux apprécier la trajectoire de la balle avant et après qu'on l'ait frappée, surtout celle de balles ayant une trajectoire irrégulière comme la balle à trajectoire ondulatoire et la balle chargée passant dans le champ électrique.

# Répartition finale des tâches

|  |  |  |
| --- | --- | --- |
|  | Tâches informatiques accomplies | Tâches scientifiques accomplies |
| Olivier | * Implémentation du système de sauvegarde: écriture et lecture d'un objet dans un fichier pour chacun des modes de jeu. * Création des menus: utilisation d'une image personnalisée pour chaque état des boutons et des cases à cocher, création du menu principal, du menu de pause et du menu d'aide. * Application d'écouteurs personnalisés pour gérer des comportements comme le changement d'état de jeu et la gestion des configurations de jeu * Création des règles de jeu pour le mode standard (nombre d’essai, les limites du HOMERUN et du HORS-JEU) | * Implémentation de calculs sur les vecteurs: addition, multiplication, module, etc. * Implémentation de la force de gravité et donc de la trajectoire balistique de la balle. * Implémentation de l’impulsion par la batte sur la balle. * Amélioration et précision importantes dans la collision des objets de jeu |
| Alexandre | * Implémentation du dessin des vecteurs. * Création des modes de jeu: implémentation des comportements et du dessin des objets de jeu comme les plateformes, les obstacles, les balles, la batte, les écouteurs de réinitialisation des balles et le pointage. * Implémentation de la caméra du jeu et le mouvement manuel de la caméra avec un écouteur de clavier * Implémentation des menus de paramètre en mode libre, tels que le sélecteur de balle avec les différents types de balle et le sélecteur de paramètres comprenant un sélecteur pour la vitesse, pour la longueur d'onde, la période et l'amplitude de la balle ondulatoire et pour la charge de la balle chargée. | * Implémentation de la force électrique et du champ électrique * Implémentation de la force de frottement * Implémentation du comportement de la balle ondulatoire (utilise la théorie des ondes électromagnétiques) * Création de la méthode de détection de la collision des objets de jeu |

# Complexité du projet

La vaste majorité du travail de départ se concentrait sur le squelette de l'application et cette base était très abstraite. Il fallait se représenter le produit final et implémenter un comportement général sur lequel tous les objets allaient dépendre. Notre application se base sur une structure typique aux jeux vidéo, les états de jeu. Ceux-ci représentent les différents modes d'interaction possible selon le contexte que l'on peut retrouver dans différents menus et dans le jeu lui-même ou ses différentes sous-catégories. Il était donc important de créer une structure convenable pour que nous puissions changer l'état qui doit être exécuté plutôt que de gérer plusieurs processus indépendamment. Une autre difficulté rencontrée était la coordination entre notre processus et le processus graphique de la librairie Swing. En effet, il est hautement déconseiller d'exécuter des calculs intensifs et de longue durée sur le processus graphique, car toute l'application se mettrait à bloquer. Cependant, plusieurs évènements nécessitaient soit le clic d'un bouton, soit le maintien d'un bouton de la souris. Puisque ces évènements se passent sur le thread graphique, il fallait trouver un moyen efficace de coordonner l'utilisation des deux processus sans causer des erreurs de concurrence. Du côté physique, frapper la balle fût un certain défi, car le professeur que nous avons eu en mécanique n'appréciait pas trop la notion d'impulsion. Cependant, cette notion et ces dérivées étaient très importantes pour la collision au complet. Appliquer une impulsion selon le mouvement de la batte, produit par la souris, était dure à concevoir, car les deux objets devraient subir autant cette impulsion, mais à cause du moment d'inertie qui ne peut être représenté dans ce système d'axe, la batte transfert toute son énergie et son inertie à la balle. Cependant, nous avons trouvé un moyen de l'appliquer sans utiliser le moment tout en appliquant toute la force sur la balle et en appliquant le coefficient de restitution pour faire réduire un peu la vitesse selon le principe de collision inélastique.

# Algorithmes externes

* Pour notre classe de vecteur en deux dimensions, nous avons utilisé une classe de vecteur en trois dimensions qui était disponible dans le projet sur le *Raytracing* qui nous a été donné en laboratoire en cours d'onde et physique moderne. Ceci nous a permis d'avoir une exemple précis sur l'implémentation d'un vecteur dans l'environnement virtuel et nous a permis d'économiser du temps pour se concentrer sur le reste du projet.
* Pour la création de notre plateforme et de nos obstacles, nous avons décidé les créer à partir des pixels d'une image où chaque pixel représente un bloc 10m2. Ceci nous empêche bien évidemment de construire des obstacles aux formes irrégulières, mais le manque de temps et un bug que nous avons surnommé "le mur inexistant" nous a poussés à conserver cette petite tentative farfelue.

# Conclusion

En conclusion, nous sommes très satisfaits du résultat, malgré un petit manque de temps. Si nous avions eu 50 heures de plus, nous aurions certainement ajouté des obstacles un peu plus complexes, nous aurions rendu l'application un peu plus jolie et nous aurions ajouté des niveaux différents, peut-être même un petit générateur aléatoire. Sans être trop optimistes, l'effet de l'air, que nous avions pensé au début du projet, aurait peut-être été réalisable. Cependant, pouvons nous considérer satisfaits et même fiers de la tâche complexe que nous venons d'accomplir. Il ne s'agit qu'un début. Il serait cependant mieux de trouver une meilleure librairie que Swing pour ce genre de projet…

**1096 mots**