Rapport de Projet

Réalisation du jeu *Space Invaders*

Raja Ouali, Bochra Arbia

1. Introduction

Ce projet consiste en la réalisation d'un clone du célèbre jeu d'arcade *Space Invaders*. L'objectif pédagogique principal était de concevoir une application robuste en langage C, tout en assurant une séparation stricte entre la logique métier du jeu et l'interface utilisateur.

Le défi technique majeur a été le développement d'un moteur de jeu unique (le **Modèle**) capable d'être piloté et affiché par deux interfaces radicalement différentes (les **Vues**) : une interface en mode texte via *Ncurses* et une interface graphique moderne basée sur *SDL3*.

Cette contrainte a guidé l'ensemble des choix d'architecture et de conception du projet.

2. Architecture logicielle : le pattern MVC

Afin de garantir la modularité, la maintenabilité et l'évolutivité du code, nous avons adopté l'architecture **Modèle–Vue–Contrôleur (MVC)**. Cette structure permet de modifier l'affichage sans impacter la logique interne du jeu.

2.1 Le Modèle

Le modèle constitue le cœur du programme. Il est totalement indépendant de toute bibliothèque graphique et n'inclut ni `<ncurses.h>` ni `<SDL3/SDL.h>`.

Il regroupe :

- les structures de données (joueur, ennemis, projectiles, bunkers),
- la gestion de l'état du jeu (vies, score, niveau courant),
- la logique physique (déplacements et calcul des collisions).

Cette indépendance garantit un comportement identique du jeu quelle que soit la vue utilisée.

2.2 La Vue

La vue est responsable de l'affichage et de la capture des entrées utilisateur. Pour gérer le polymorphisme en C, un langage non orienté objet, nous avons mis en place une interface abstraite reposant sur des pointeurs de fonctions.

Deux implémentations distinctes ont été développées :

- **Vue Ncurses** : affichage en caractères ASCII dans le terminal, légère et idéale pour le débogage.
- **Vue SDL** : affichage graphique utilisant des sprites, des textures et des polices TrueType (TTF).

2.3 Le Contrôleur

Le contrôleur orchestre la boucle principale du jeu (*Game Loop*). À chaque itération, il :

- récupère les commandes abstraites issues de la vue (CMD_LEFT, CMD_SHOOT, etc.),
- met à jour le modèle en fonction du temps écoulé,
- déclenche le rafraîchissement de l'affichage,
- régule le framerate afin d'assurer une vitesse constante.

3. Gestion de la mémoire et qualité du code

La gestion manuelle de la mémoire étant un point critique en C, une attention particulière a été portée à la conception et à la libération des ressources.

4. Points critiques et solutions techniques

4.1 Difficulté dynamique

Afin d'éviter un gameplay monotone, une difficulté dynamique a été implémentée. Le contrôleur surveille le nombre d'ennemis encore actifs : plus ce nombre diminue, plus leur vitesse de déplacement augmente.

Le délai entre deux mouvements est recalculé à chaque ennemi détruit, créant une montée progressive de la tension et reproduisant fidèlement la sensation du jeu original de 1978.

4.2 Installation de SDL3

La principale difficulté du projet a été l'installation et la configuration de la bibliothèque *SDL3*, qui n'est pas disponible par défaut sur la plupart des systèmes.

5. Conclusion

Ce projet a permis de consolider nos compétences en programmation C avancée. La mise en œuvre de l'architecture MVC et la gestion automatisée des dépendances externes nous ont confrontés à des problématiques proches du développement logiciel professionnel.

Le jeu est aujourd'hui fonctionnel, stable (sans fuite mémoire) et offre une expérience utilisateur fluide aussi bien en mode texte qu'en mode graphique.