PLESSY Vincent TP1

INCOGNITO

2024/2025

Rapport de développement et documentation

1. Documentation utilisateur

1.1. Introduction

Incognito est un jeu de plateau stratégique pour deux joueurs sur un plateau de 5x5 cases. Chaque joueur contrôle des chevaliers et un espion, et le but est de capturer le château adverse avec l'espion ou de découvrir l'espion de l'adversaire.

Le jeu offre deux modes d'affichage :

- **Mode texte** en ASCII (avec l'option `-a`),

- **Mode graphique** avec la bibliothèque **MLV** (avec l'option `-g`).

1.2. Fonctionnalités principales

Modes d'affichage :

- **ASCII** : Affichage textuel lorsque l'option `-a` est spécifiée en ligne de commande.

- **Graphique** : Interface graphique via MLV lorsque l'option `-g` est spécifiée.

Sauvegarde et chargement :

- **Sauvegarde** : L'option `-s <fichier>` permet d'enregistrer la partie en cours.

- **Chargement** : L'option `-c <fichier>` permet de charger une partie enregistrée.

Déroulement du jeu :

Les joueurs jouent chacun leur tour. Lors d'un tour, le joueur peut déplacer un pion ou interroger une pièce adverse pour découvrir son espion. Le jeu se termine lorsqu'un château est capturé ou lorsqu'un espion est découvert.

1.3. Instructions de jeu

Lancement du jeu :

- **Mode texte** : ./incognito -a

- **Mode graphique** : ./incognito -g

Sauvegarde et chargement d'une partie :

- **Sauvegarder** : ./incognito -a -s sauvegarde.inco

- **Charger** : ./incognito -g -c sauvegarde.inco

1.4. Règles du jeu

Objectifs :

- Capturer le château adverse : Si l'espion d'un joueur atteint le château ennemi, ce joueur gagne. Les châteaux sont situés aux cases [0,4] pour le joueur noir et [4,0] pour le joueur blanc.

- Démasquer l'espion adverse : Un joueur gagne s’il interroge et découvre l’espion de l'adversaire.

Déplacements :

Les pions se déplacent d’une case en ligne droite ou en diagonale. Les joueurs ne peuvent pas se déplacer sur leur propre château.

Interrogation :

Un pion peut interroger un pion adverse adjacent. Si le pion interrogé est un espion, l'adversaire perd. Sinon, le pion interrogateur est éliminé.

2. Documentation technique

2.1. Structure du code

Structures principales :

- `Pion` : Représente chaque pion avec son type (`CHEVALIER` ou `ESPION`) et sa couleur (`BLANC` ou `NOIR`).
- `Case` : Représente les coordonnées (x, y) sur le plateau.
- `Mouvement` : Contient les cases de départ et d'arrivée d'un mouvement ainsi qu'une indication si c’est une interrogation.
- `Jeu` : Structure principale contenant le plateau, le joueur en cours, l'historique des mouvements, et l'état de la partie.

Fonctions principales :

- Initialisation : `**initialiser_jeu()**` prépare le plateau avec les pions de chaque joueur à leurs positions initiales.

- Affichage :

- `**afficher_plateau_ascii()**` pour un affichage textuel en mode console.
- `**afficher_plateau_graphique()**` pour un affichage en mode graphique avec **MLV**.

- Vérification de mouvement : `**est_mouvement_valide()**` vérifie si le déplacement respecte les règles du jeu.

- Interrogation : `**interroger_piece()**` permet à un joueur d’interroger un pion adverse et vérifie si le pion ciblé est un espion.

- **Interaction utilisateur :**

- `**demander_action_ascii()**` pour interagir en mode texte.
- `**demander_action_graphique()**` pour gérer les clics de l’utilisateur en mode graphique.

- **Sauvegarde et chargement :**

- `**sauvegarder_partie()**` enregistre l’état du jeu dans un fichier.
- `**charger_partie()**` restaure une partie depuis un fichier sauvegardé.

- **Libération des ressources** : `**liberer_jeu()**` libère la mémoire allouée aux structures de jeu.

2.2. Détails d'implémentation

Alternance des joueurs : La structure `Jeu` maintient l'état du joueur en cours, qui est alterné après chaque tour.

Validation des mouvements : La fonction `**est_mouvement_valide()**` vérifie les règles du jeu, y compris les restrictions pour le château et les mouvements en ligne droite ou en diagonale.

Gestion graphique : En mode graphique, les pions sont dessinés sur le plateau avec un code couleur (blanc pour les blancs et noir pour les noirs), et les espions sont visuellement identiques aux autres pions.

3. Méthode de travail

3.1. Méthodologie mise en œuvre

J'ai adopté une approche Agile avec des sprints hebdomadaires et des rétrospectives pour ajuster les priorités.

Outils :

- **Trello** pour la gestion des tâches.

3.2. Répartition des tâches

- Implémentation des règles de jeu et gestion des tours.

- Création de l'interface graphique et gestion des interactions avec **MLV**.

- Sauvegarde et chargement des parties, organisation du code pour faciliter la maintenance.

4. Difficultés rencontrées et résolutions

- **Compatibilité de MLV** : Des problèmes d'installation sous Windows ont été résolus par l'installation de dépendances et une configuration du makefile.

- **Affichage des pions en mode graphique** : Les coordonnées des cases ont été ajustées pour corriger la détection des clics.

- **Gestion de l'alternance des joueurs** : Un bug initial dans `**changer_joueur()**` a été corrigé pour assurer un changement de joueur fluide après chaque action.

5. Potentielles Améliorations Réalisées

5.1. Interface graphique plus intuitive

Optimisation des réactions immédiates aux actions des joueurs.

5.2. Améliorations du mode texte

Le mode texte a été optimisé pour afficher des informations claires après chaque action. Le joueur sait immédiatement si son mouvement ou son interrogation a réussi ou échoué, et le jeu alterne automatiquement les tours entre les joueurs.

5.3. Sauvegarde robuste

Le système de sauvegarde a été revu pour stocker toutes les informations nécessaires à la reprise d'une partie, y compris les positions des pions et l'état actuel du jeu. Cela permet une reprise fluide des parties sauvegardées.