Sommaire exécutif

Ce rapport détaille mon stage de fin d'études en intelligence artificielle, axé sur le développement d'un jeu vidéo intégrant des techniques avancées d'IA. Le projet, d'une durée de 300 heures, visait à créer un prototype de jeu en utilisant Unity avec Entity Component System (ECS) une programmation orienté données et en implémentant l'apprentissage par renforcement en continue pour les comportements ennemis.

Les objectifs principaux comprenaient la maîtrise d'Unity ECS, l'intégration de l'apprentissage par renforcement, et l'optimisation des performances du jeu. Les réalisations clés incluent la configuration et l'implémentation des mécaniques de base du jeu avec ECS, et le développement d'un système d'IA adaptatif pour les ennemis.

Un défi majeur a été la transition vers la programmation orientée données d'ECS, nécessitant une refonte complète de l'approche de développement. Cette difficulté a été surmontée grâce à une formation intensive et une restructuration du code.

Le projet a abouti à un prototype fonctionnel démontrant l'efficacité de l'IA dans un environnement de jeu dynamique. Cette expérience a considérablement renforcé mes compétences en développement de jeux et en IA appliquée, tout en offrant une base solide pour de futures innovations dans ce domaine.

Ce stage a été une opportunité inestimable d'appliquer des concepts théoriques d'IA à un projet concret.

Introduction

Dans le cadre de ma formation en spécialisation technique en intelligence artificielle au Cégep de Ste-Foy, j'ai entrepris un stage de fin d'études axé sur le développement d'un jeu vidéo intégrant des techniques avancées d'IA. Ce projet personnel ambitieux visait à fusionner ma passion pour le développement de jeux avec mon expertise croissante en intelligence artificielle.

Le projet consistait à développer un prototype de jeu vidéo en utilisant Unity avec Entity Component System (ECS), une approche de programmation orientée données conçue pour optimiser les performances. L'objectif principal était d'intégrer l'apprentissage par renforcement pour créer des comportements ennemis adaptatifs et dynamiques, offrant ainsi une expérience de jeu plus immersive et stimulante.

Les objectifs spécifiques du stage comprenaient :

- Maîtriser Unity ECS et ses principes de programmation orientée données
- Implémenter un système d'apprentissage par renforcement en continu et en jeu pour les comportements ennemis
- Optimiser les performances du jeu en tirant parti des avantages d'ECS
- Développer des compétences en gestion de projet et en développement itératif

Ce projet s'inscrit dans une tendance croissante de l'industrie du jeu vidéo à intégrer des technologies d'IA avancées pour améliorer l'expérience de jeu. L'utilisation de l'apprentissage par renforcement pour créer des adversaires plus intelligents et adaptatifs représente une frontière passionnante dans le développement de jeux, offrant des défis plus engageants et des expériences plus personnalisées aux joueurs.

La méthodologie adoptée pour ce projet a suivi une approche de développement itératif, avec des cycles courts de développement, de test et d'optimisation. Cette approche a permis une adaptation rapide aux défis techniques rencontrés et une amélioration continue du prototype.

Ce rapport est structuré en plusieurs sections clés. Après cette introduction, nous explorerons le contexte organisationnel et la planification du projet. Ensuite, nous détaillerons

l'environnement technologique et méthodologique, suivi d'une description approfondie de la réalisation du projet, y compris les défis rencontrés et les solutions apportées. Nous aborderons ensuite l'expérience professionnelle et personnelle acquise, avant de conclure avec une réflexion sur l'encadrement reçu et les perspectives futures.

À travers ce rapport, je vise à démontrer non seulement les compétences techniques acquises, mais aussi la valeur de l'application pratique de l'IA dans le développement de jeux vidéo. Cette expérience représente une étape cruciale dans ma formation et pose les bases d'une carrière à l'intersection passionnante du développement de jeux et de l'intelligence artificielle.

Environnement technologique et méthodologie

L'environnement technologique de ce projet a été soigneusement sélectionné pour optimiser le développement et l'intégration de l'IA dans le jeu. Le cœur du projet repose sur Unity avec

son Entity Component System (ECS), offrant une base solide pour la programmation orientée données et l'optimisation des performances.

L'environnement de développement comprend un système d'exploitation Ubuntu, choisi pour sa stabilité et sa compatibilité avec les outils de développement d'IA. Le moteur de jeu utilisé est Unity 2022.3 LTS avec le package Entities pour ECS. La gestion de version est assurée par Git et GitHub, permettant un contrôle de version efficace et une collaboration facilitée. Le langage de programmation est exclusivement C#, sans utilisation de bibliothèques externes d'IA.

La méthodologie adoptée pour l'implémentation de l'IA s'est déroulée en deux phases principales : l'implémentation initiale d'un algorithme de Q-learning basé sur une grille, suivie d'une transition vers un système de Deep Reinforcement Learning (DRL).

Pour le Q-learning basé sur une grille, l'algorithme a été implémenté en C# pur, intégré dans la structure ECS de Unity. Les principales étapes comprenaient la définition de l'état, la représentation de l'environnement de jeu sous forme de grille, l'implémentation d'une table Q pour stocker les valeurs Q, l'utilisation d'une politique ε-greedy pour équilibrer entre exploration et exploitation, et la mise à jour de la table Q selon la formule de Q-learning standard. L'implémentation en ECS a nécessité la création de composants spécifiques pour stocker les données d'état et d'action, ainsi que des systèmes pour gérer la logique de mise à jour de la table Q.

Pour améliorer la capacité d'adaptation et la généralisation de l'IA, une transition vers un système DRL a été effectuée, toujours implémenté en C# natif sans bibliothèques externes. Un réseau neuronal multicouches a été implémenté en C# pur, remplaçant la table Q par une fonction d'approximation basée sur le réseau neuronal. Un replay buffer a été mis en place pour le stockage et l'échantillonnage des expériences, et l'algorithme DQN (Deep Q-Network) a été adapté pour utiliser le réseau neuronal.

L'intégration du DRL dans Unity ECS a nécessité une approche innovante. Des composants ECS ont été créés pour stocker les poids du réseau neuronal et les données d'expérience, et des systèmes ECS ont été développés pour la prédiction, l'apprentissage et la mise à jour du réseau neuronal. L'optimisation des performances a été réalisée grâce à l'utilisation des Jobs System et Burst Compiler d'Unity pour paralléliser les calculs du réseau neuronal.

Un aspect crucial du projet était l'implémentation d'un apprentissage continu pendant le jeu. La collecte de données en temps réel capturait les interactions joueur-ennemi pendant le gameplay. La mise à jour asynchrone, permettait d'effectuer l'apprentissage en arrière-plan

sans impacter les performances du jeu. Une adaptation dynamique ajustait en temps réel les paramètres du réseau neuronal basé sur les nouvelles expériences de jeu.

Cette approche méthodologique, combinant des techniques classiques de Q-learning avec des méthodes avancées de DRL, le tout intégré dans l'architecture ECS de Unity, a permis de créer un système d'IA adaptatif et performant, capable d'évoluer en temps réel pendant le jeu. Les défis rencontrés, notamment dans l'optimisation des performances et la gestion de la complexité du code, ont été surmontés grâce à une utilisation judicieuse des fonctionnalités d'Unity ECS et une conception soignée de l'architecture du système d'IA.

Voici la planification original avec le resultat :

PHASE DU PROJET	HEURES ATTRIBUÉES	DÉTAILS	RÉSULTAT
Configuration et Planification	35	Installation/configuration d'Ubuntu, Unity, Git, GitHub; Planification	complété
Développement Initial des mécaniques du jeu	80		
- Familiarisez-vous avec les concepts ECS	8	Entités, Composants, Systèmes	complété
- Implémentez le mouvement basique du joueur	8	Création entité joueur, système d'entrée	complété
- Générez des ennemis	10	Entités ennemies, système de génération	complété
- Implémentez le système de combat	10	Composant d'attaque, système de collision	complété
- Système de power-ups/améliorations	10	Entités de power-up, système de collecte	Non complété
- Implémentez des mécaniques de survie	10	Système de minuterie, système de score	
- Logique de mort et de redémarrage	8	Système de santé, option de redémarrage	complété

- Interface utilisateur basique pour le retour d'information	5	Systèmes UI pour afficher la santé, le score/temps, les power-ups	Seulement le system de santé
- Système audio simple	5	Composants et systèmes audio	Non complété
Phase d'Apprentissage par Renforcement	135		
- Établir la Fondation	18	Définition de l'environnement, des espaces d'actions	complété
- Implémenter un Agent Basique	18	Architecture de l'agent initial, boucle d'entraînement	complété
- Simuler, Collecter des Données et Entraînement Initial	18	Collecte de données, premier entraînement	complété
- Introduire la Dynamique Multi-Agents	18	Adaptation pour le multi-agent	complété
- Implémenter des Algorithmes Plus Complexes	18	Sélection et implémentation d'algorithmes avancés	complété
- Infrastructure de Formation Continue	18	Systèmes pour la collecte continue de données	complété
- Évaluation, Itération et Affinement	9	Surveillance de la performance, ajustements	Non complété
- Considérations Éthiques et Consentement des Joueurs	9	Mesures de transparence et de consentement	Non complété
- Améliorations Avancées	9	Diversification des comportements, optimisation	complété
Test et Optimisation	35		
- Playtesting	12	Organisation de sessions de jeu, identification des problèmes	Non complété
- Analyse des performances	12	Profilage du jeu, identification des goulots d'étranglement	Non complété

- Optimisation IA/jeu avec ECS	11	Révision des systèmes ECS, ajustement des modèles d'IA	complété
Finalisation et Préparation à la Production	10		
- Finalisation des fonctionnalités	3.5	Peaufinage, préparation pour la production	Non complété
- Préparation des documents	3.5	Documentation technique, supports de présentation	En cours
- Tests de performance finaux et ajustements	3	Tests de charge, ajustements finaux	complété
Planif ajouté			
-licenses		Ajouter une licenses approprier au github	complété

Enjeux et défis du projet

Le projet a abouti à la création d'une démo fonctionnelle d'un niveau en 3D, mettant en scène des ennemis contrôlés par un système de Deep Reinforcement Learning (DRL) capable d'exécuter diverses actions distinctes. Les réalisations principales incluent la conception et la modélisation d'un niveau unique mais complexe, l'implémentation de la physique et des collisions en utilisant Unity ECS, ainsi que la création d'un système de d'obstacles pour varier l'expérience de jeu.

Un système de combat fluide et réactif simple a été développé, ainsi qu'un système de santé et de dégâts pour le joueur et les ennemis. Pour l'intelligence artificielle des ennemis, un réseau neuronal en C# pur a été implémenté, accompagné d'un système de perception permettant aux ennemis d'analyser leur environnement et de prendre des décisions basées sur les sorties du DRL.

Un système d'apprentissage continu a été mis en place, comprenant un mécanisme de collecte de données en temps réel et un système de mise à jour asynchrone du réseau neuronal. Un replay buffer optimisé pour ECS a également été créé. L'interface utilisateur a été conçue de manière minimaliste, affichant les informations essentielles et incluant un système de feedback visuel pour les actions de l'IA.

Le projet a présenté plusieurs défis significatifs, notamment la complexité du code due à l'implémentation d'un DRL en C# pur sans bibliothèques externes, nécessitant une architecture modulaire stricte. L'optimisation des performances pour exécuter le DRL en temps réel pour plusieurs ennemis simultanément a été réalisée grâce à l'utilisation intensive du Job System d'Unity et d'un système de mise à jour asynchrone pour l'apprentissage.

L'absence de métriques de performance stables, due à l'apprentissage continu redémarrant à chaque partie, et la gestion de la mémoire, avec le replay buffer et les données du réseau neuronal consommant beaucoup de mémoire, constituent les principaux défis restants à surmonter. Ces défis représentent des occasions précieuses de développer des compétences cruciales en résolution de problèmes et en pensée créative. La nécessité de concevoir des solutions innovantes, telles que le système d'apprentissage asynchrone et les outils de visualisation en temps réel, permettrait d'élargir considérablement mon arsenal technique.

Les bénéfices obtenus du projet incluent une compréhension approfondie du DRL, la capacité accrue à concevoir et implémenter des systèmes d'IA complexes sans dépendre de bibliothèques externes, et le développement d'une expertise dans l'utilisation d'Unity ECS

pour des projets à haute performance. Le projet a également innové dans le gameplay en créant une expérience de jeu unique où les ennemis évoluent de manière organique au fil des parties, offrant un potentiel de rejouabilité accrue. Enfin, le projet a posé les fondations pour de futures recherches dans le domaine de l'IA appliquée aux jeux et a développé des compétences en résolution de problèmes techniques complexes.

Conclusion

Ce stage, axé sur le développement d'un jeu vidéo intégrant des techniques avancées d'intelligence artificielle, a été une expérience immensément enrichissante et formatrice. Au terme de ces 300 heures de travail intensif, plusieurs réflexions s'imposent.

Tout d'abord, la réalisation de ce projet a démontré la puissance et le potentiel de l'apprentissage par renforcement profond dans le contexte du développement de jeux vidéo. L'implémentation d'un système DRL entièrement en C# natif, intégré dans l'architecture ECS d'Unity, a non seulement été un défi technique considérable, mais aussi une opportunité unique d'approfondir ma compréhension des mécanismes sous-jacents de l'IA.

Ce projet a également mis en lumière l'importance de l'équilibre entre l'innovation technique et l'expérience utilisateur. La création d'ennemis contrôlés par IA, capables d'apprendre et de s'adapter en temps réel, ouvre des possibilités fascinantes pour l'avenir des jeux vidéo. Cependant, cela soulève également des questions importantes sur la façon de maintenir un gameplay équilibré et engageant face à des adversaires en constante évolution.

Sur le plan personnel, ce stage a renforcé ma passion pour l'intersection entre le développement de jeux et l'intelligence artificielle. Il m'a permis de consolider mes compétences techniques tout en développant des qualités essentielles telles que la persévérance, l'adaptabilité et la gestion de projet.

En regardant vers l'avenir, je suis convaincu que les connaissances et l'expérience acquises au cours de ce stage seront inestimables. Les possibilités d'application et d'expansion des techniques développées ici sont vastes, allant de l'amélioration des systèmes de NPCs dans les jeux à grande échelle, à l'exploration de nouvelles formes de narration interactive guidée par l'IA.

En conclusion, ce stage a non seulement atteint ses objectifs initiaux, mais les a dépassés en offrant une expérience d'apprentissage profonde et multidimensionnelle. Il a renforcé ma conviction que l'avenir du développement de jeux est intimement lié aux avancées en intelligence artificielle. Avec les compétences et les connaissances acquises, je suis enthousiaste à l'idée de contribuer à cette évolution passionnante de l'industrie du jeu vidéo.