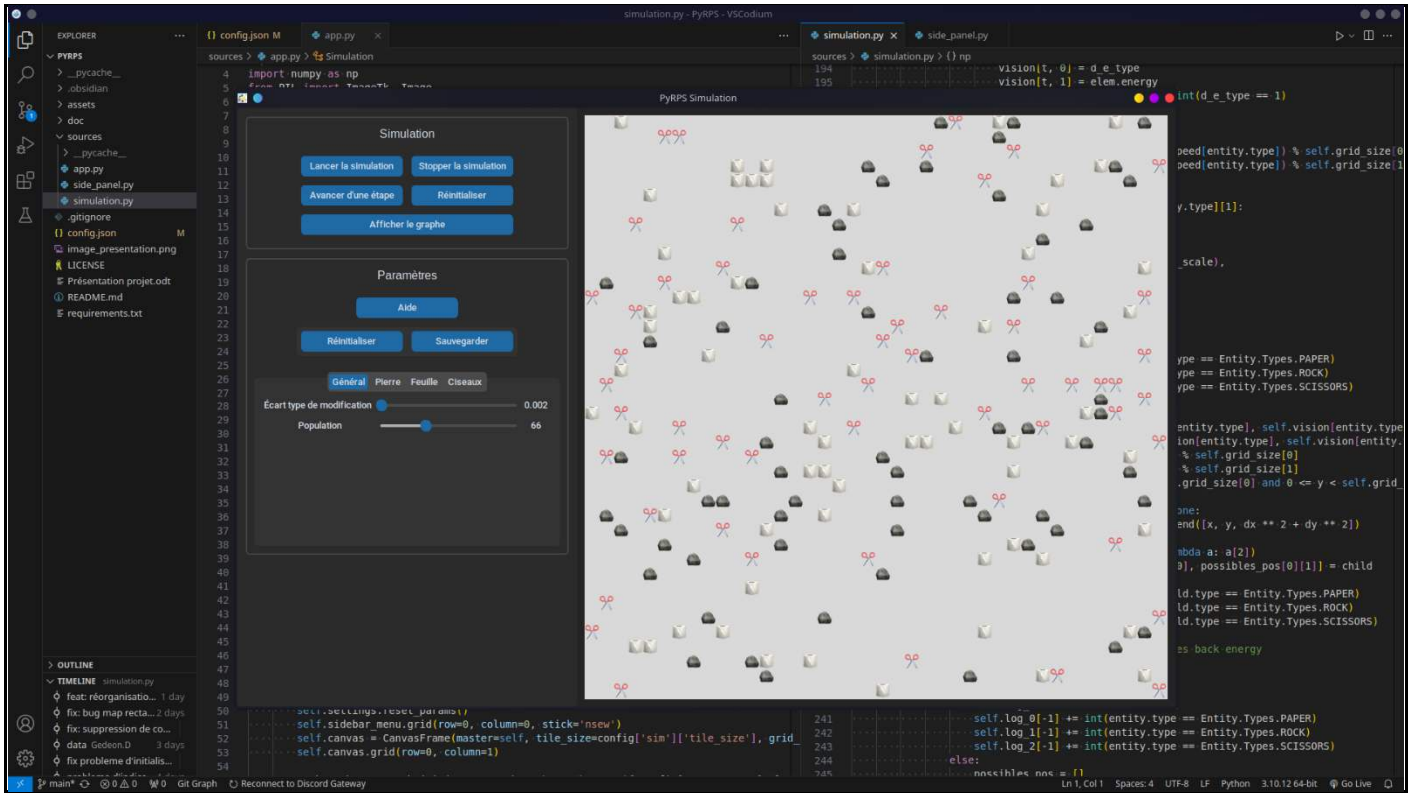
LES TROPHÉES NSI

ÉDITION 2024

DOSSIER DE CANDIDATURE

PRÉSENTATION DU PROJET



NOM DE VOTRE PROJET :	PYRPS
MEMBRE DE L'ÉQUIPE :	JULES COURTIADE-VANIN
MEMBRE DE L'ÉQUIPE :	GÉDÉON DEVEAUX
NIVEAU D'ÉTUDE :	PREMIÈRE
ÉTABLISSEMENT SCOLAIRE :	LYCÉE DU GRANIER
ENSEIGNANTE/ENSEIGNANT DE NSI :	NOÉMIE EXARTIER

> PRÉSENTATION GÉNÉRALE :

PyRPS est un simulateur de pierre-feuille-ciseaux à grande échelle, propulsé par l'intelligence artificielle et les réseaux de neurones. Regardez comment 3 populations d'entités évoluent dans un environnement commun, avec les règles du jeu « poule-renard-vipère » (chaque population a une population en prédateur et l'autre en proie).
Ce projet est né d'une vidéo drôle vue sur YouTube, ainsi que de l'envie d'établir définitivement qui est le meilleur entre la pierre, la feuille et les ciseaux (le puit est exclu ici).
L'objectif principal reste néanmoins sérieux, il est de pouvoir étudier et analyser le comportement de ces créatures dans leur environnement face à leurs proies et à leurs prédateurs et de comprendre quels mécanismes de défense et d'attaque elles mettent en place tout au long de leur évolution au cours du temps.

> ORGANISATION DU TRAVAIL :

Lors de la première phase de développement du projet, où nous avons itéré sur la structure et l'articulation des différents éléments (backend de la simulation et interface utilisateur), Gédéon s'est occupé de la partie « backend ». Il a créé la simulation, les réseaux de neurones ainsi que les entités et leurs différentes caractéristiques. Jules a développé la partie « frontend », le design de l'interface, l'affichage des entités sur la carte, la gestion des interactions avec les boutons de contrôle de la simulation, la création de la simulation et l'affichage des statistiques.
Lors de la seconde phase de développement, nous avons majoritairement résolu des bugs ainsi que clarifié et restructuré le code pour en simplifier la compréhension, et pour correspondre aux critères d'organisation des fichiers. Nous avons aussi continué à développer des fonctionnalités, mais cela est resté au second plan, derrière le travail pour constituer le dossier et livrer une application fonctionnelle et intuitive à utiliser. Chacun a travaillé aussi bien sur le « backend » que sur le « frontend ».
Nous avons principalement travaillé en dehors de l'établissement, quelques fois lors des cours de NSI mais cela était plutôt rare étant donné les autres projets que nous menons lors des cours.

Pour développer ce projet, nous avons utilisé l'IDE Visual Studio Code et son dérivé Codium, car nous le trouvons très complet (autocomplétions, coloration syntaxique) et nous avons l'habitude de l'utiliser dans nos projets personnels. De plus, Visual Studio Code comprend nativement une interface graphique pour Git, un logiciel de gestion de versions. A ce propos, nous avons utilisé Git ainsi qu'un serveur Git personnel avant de migrer sur GitHub pour pouvoir collaborer efficacement à distance, et avancer chacun sur nos objectifs tout en ayant toujours les dernières mises à jour de l'autre. GitHub nous a aussi permis de tracer les bugs et de les résoudre grâce aux issues, de développer et d'intégrer facilement les nouvelles fonctionnalités grâces aux branches et aux pull requests, ainsi que de suivre le développement du projet, et de ce qu'il nous restait à faire.

LES ÉTAPES DU PROJET :

La première étape fut l'idée. Nous n'étions vraiment pas inspirés par le thème du sport, donc nous avons cherchés chacun de notre côté, mais aucune idée ne respectait assez les restrictions (langages interdits ou majorité de HTML/JavaScript). Finalement, après avoir vu une vidéo simulant une partie de pierre-feuille-ciseaux, nous avons choisi de partir de cela. Etant tous les deux fascinés par l'IA, nous avons décidé d'en inclure un peu dans ce projet, et PyRPS était né.

Nous avons commencé par une première version utilisant PyGame pour afficher les entités, qui n'étaient alors que des pixels rouges, verts ou bleus en fonction du type et de luminosité variable en fonction de la vie. Petit à petit nous avons ajouté des fonctionnalités telles que les attaques, les collisions et le fait que la carte n'est pas de bord (téléportation d'un côté à un autre si une entité sort).

Nous avons ensuite surtout résolu des bugs et chercher quelles fonctionnalités apporter pour améliorer l'expérience. Une première piste fut d'utiliser Cython, qui permet de compiler du code source Python en module C pour accélérer la simulation et apporter plus de fluidité. Malheureusement l'implémentation et la compilation étaient trop compliqué par rapport à la fluidité apporté pour que cela vaille vraiment le coup.

L'idée suivante fut d'améliorer l'interface utilisateur et de permettre plus de contrôle sur la simulation, car la version utilisant PyGame n'était pas très pratique. Nous avons donc cherché comment faire des interfaces graphiques en Python, et nous avons regardé du côté de TkInter. La première version de l'UI était plutôt moche, car TkInter n'est pas très intuitif au premier abord, et les éléments graphique ne sont pas très modernes. Nous avons donc fait encore quelques recherches, et découvert la librairie CustomTkinter ajoutant une touche de modernité et une petite couche d'abstraction à TkInter, facilitant le développement et relançant la motivation de faire une UI belle et pratique.

Nous avons entre temps cherché une alternative à Cython pour fluidifier la simulation, et nous sommes tombés sur Taichi, une librairie permettant de faire de la parallélisation en Python. Malheureusement, cela ne fonctionnait pas ou très peu, nous avons donc à nouveau abandonné cette idée.

Une fois la nouvelle UI arrivée à maturité, nous sommes entrés dans la phase 2 du projet, ou nous avons majoritairement résolu des bugs et fait quelques optimisations du code. Le travail principal a été de rendre l'interface plus complète, plus belle et plus pratique à utiliser.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Au moment du dépôt du projet, celui-ci est presque terminé. Toutes les fonctionnalités majeures ont été implémentées, tous les bugs connus ont été résolus.
Pour vérifier l'absence de bugs, nous avons simplement testé l'application après chaque nouvelle modification, pour vérifier que la simulation et l'interface avaient le comportement attendu.
Des fonctionnalités annexes sont toujours en développement, tel que la création d'une autre IA ayant pour but de trouver les meilleurs paramètres pour la simulation.

> OUVERTURE :

Les nouvelles fonctionnalités à développer à moyen terme pourraient être :

- Une interface un peu plus intuitive / ergonomique du projet en général
- Créer une interface pour l'apprentissage et l'optimisation des paramètres de la simulation
- Ajout de parallélisation pour augmenter les performances
- Compiler le projet en exécutable pour différents OS avec PyInstaller

Les nouvelles fonctionnalités à long terme pourraient être :

- Transpiler le projet en un langage bas niveau pour bénéficier de performances supérieures

Un aspect regrettable du projet réside dans la durée souvent insuffisante des simulations pour susciter un réel intérêt. Afin de remédier à cette lacune, nous envisageons le développement d'une intelligence artificielle visant à optimiser les paramètres.

Au niveau de l'organisation, nous ne changerions rien, Git et Github sont des outils parfaits pour collaborer, nous avions une bonne communication et travaillons chacun de manière complémentaire.

Les compétences développées sont les suivantes :

- Workflow avec Git et Github
- Développement en collaboration (pas quelque chose dont nous avons l'habitude hors de projet en NSI)
- Création d'interface graphique en Python avec Tkinter et des librairies dérivées
- Gestion de la pression du timing de dépôt du projet
- Création et compréhension du fonctionnement et de l'apprentissage d'une IA