SPRINT GOAL

Mouvement des Agents
&
Génération du Terrain

Sprint Review

On a mis en place un placement au fur et à mesure des ticks pour les entités, avec une probabilité qui dépend du nombre d'entités déjà placées, afin de ne pas avoir un terrain qui se génère d'un coup, et le voir évoluer au fur et à mesure. Il y a aussi désormais une vérification du type du terrain avant de placer une entité afin d'éviter des situations type arbres au milieu de l'eau. Aussi les entités s'orientent désormais en fonction de leurs position sur la planète pour des raisons esthétiques.

On a mis en place un pathfinding de type "Dijkstra A*" pour les agents une fois qu'ils détectent une entité dans leurs champs de vision, et temporairement ils s'y dirigent afin de la "manger". En premier lieu nous avons tenté de le créer de nous même, mais nous avons découvert qu'il existait une version pré implémenté, de l'algorithme de pathfinding. L'on a décidé de l'utiliser afin de gagner du temps.

On a légèrement réorganisé le code et retiré plusieurs fonctions inutiles afin que cela soit plus clair.

On a amélioré la fonction d'initialisation du terrain afin de ne plus utiliser un algorithme récursif qui faisait planter le projet au dessus d'une résolution de 3. Pour cela, on a implémenter une structure File-Ensemble, c'est à dire :

- De la file on a pris les fonctions d'enfilage, de défilage, estVide()
- De l'ensemble on a pris l'unicité des éléments

On a adapté cette structure pour des cases à "collapse", donc il y a aussi un test avant d'enfiler pour savoir si une case est "undefined".
Comme on enfile d'abord les voisins d'eau puis le reste, on minimise les cas ou les cases n'ont aucun voisin défini et donc on abandonne et elle retourne tout à la fin de la file.
Cette structure nous a fait gagné un gain en mémoire considérable (Stack Overflow à plus d'erreur) et un gain de temps non-négligeable (37s -> 27s pour générer le monde avec une résolution de 4)

Ensuite, pour l'algorithme de sélection naturelle avec de l'évolution, on a préféré l'implémenter d'abord sous Python (car plus simple à implémenter/débugguer/tester) pour être sûr de l'avoir compris.
Il ne nous reste qu'à le traduire et l'adapter sur notre projet.

Sprint Retrospective

Il n'y a toujours aucun problème pour travailler en binome.

Product Increment

Numéro du dernier commit correspondant à la version finale du sprint
0c92356995748a6d16d728e30d9bc7bcfed231f5

Sprint Backlog

n°	Tâches	Séance 1	Séance 2	Séance 3	Etats
1	Trouver un sujet	-	-	-	Terminé
2	Trouver un moteur graphique	-	-	-	Terminé
3	Génération procédurale d'une planète	-	-	-	Terminé
4	Mettre en place un système de coordonnées afin de placer les agents/objets	-	-	-	Terminé
5	Déterminer les entités à placer sur la planète et les conditions de leur placement (exemple : arbres, végétation; buissons, pierres, eau, ...)	1h	0	-	Terminé
6	Evolution de la planète et génération d'entités en fonction des ressources	9h	3h	3h	En cours
7	Mettre en place le système de météorites pouvant apporter des ressources	-	-	-	Abandonné
8	Mettre en place une UI et un joueur pouvant se déplacer et contrôler le monde (exemple : faire tomber des météorites, faire apparaitre des agents, ...)	6h	5h	2h	En cours
9	Mettre en place des agents dont les caractéristiques diffèrent (régime alimentaire, conditions de survie, vitesse etc...)	3h	2	1	En cours
10	Evolution des agents	8h	6	5	En cours
11	Statistiques	2h	1h45	1h45	En cours
12	Ajouts bonus (exemple : caméra centré sur agents, amélioration de la génération de la planète, optimisation des performances (compute shaders),)	-	-	-	En attente