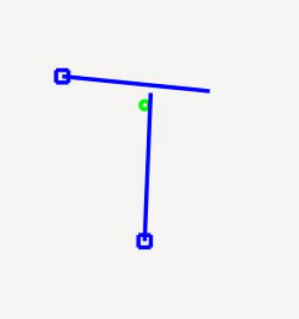
Rapport Rendu 3 :

Préambule :

Pour comprendre ce qui va suivre, il faut comprendre le fonctionnement général de la détection de collisions. Notre code, comme sous-entendu dans le paragraphe de la donnée sur epsil_zero, fait la mise à jour de chaque corail (incrémentation de l'âge avec éventuelle mort, déplacement de maximum delta_rot, éventuelle consommation d'algue avec extension de longueur, éventuelle reproduction ou création d'un nouveau segment). Puis il détecte s'il y a une collision (superposition, bords du Monde, avec lui-même, avec un autre corail) pour chaque corail, et dans ce cas il réinitialise la simulation et lit un fichier texte avec l'état de la simulation, avant toute mise à jour, et enfin change la direction de chaque corail ayant détecté une collision.

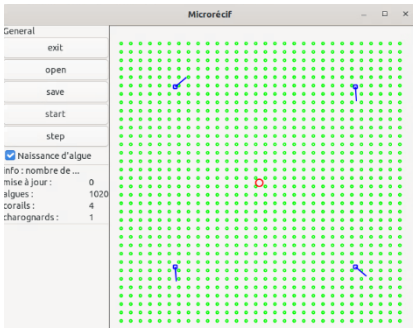
Description d'exécution pour deux fichiers de test :

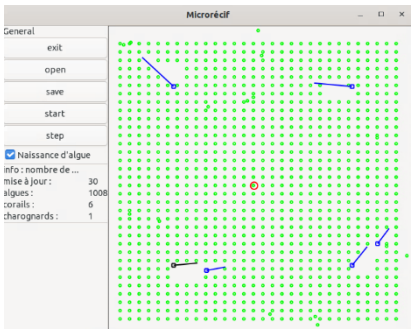
t43.txt :



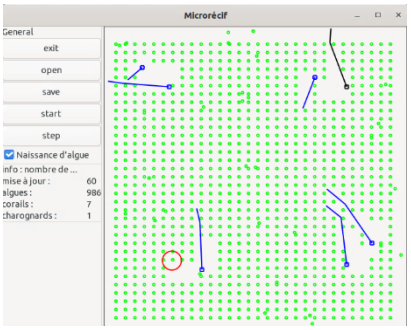
Dans le cas du test 43, la simulation ne peut pas faire de mise à jour qui ne soit pas immédiatement annulée. En effet, lors de la première mise à jour, les deux coraux tournent en TRIGO. Ainsi, le corail du bas va manger l'algue et s'étendre. Il y aura donc une détection d'erreur (collision) pour les deux coraux ce qui fait revenir la simulation dans son état précédent. A l'étape deux, les deux coraux tournent en INVTRIGO (leurs deux effecteurs s'étant collisionnés, ils ont tous les deux changé de direction de rotation). Ainsi le corail du haut va tourner sur le corail du bas et une collision va être détecté pour les deux coraux. A nouveau la simulation revient à sa situation initiale et cela à l'infini. La simulation garde donc toujours l'aspect de la capture ci-dessus.

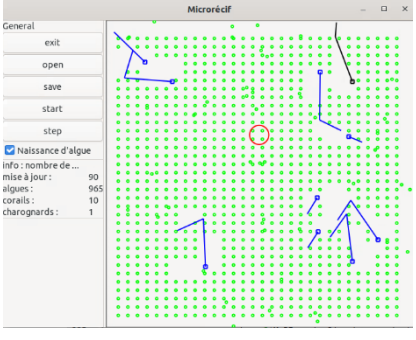
t46.txt :





A l'itération 90, nous avons 965 algues, 10 coraux et 1 scavenger, soit un total de 976 entités.





Méthodologie et conclusion :

Organisation du travail :

Pour ce projet nous avons utilisé Github pour stocker le code écrit et la VM pour coder. Lorsque l'un de nous souhaitait travailler, il prévenait l'autre de ne pas toucher au code. Une fois son travail abouti, il mettait la nouvelle version sur le repository.

Nous avons quelque fois travaillé ensemble mais l'essentiel du travail s'est fait chacun de notre côté. En effet, selon nous, il n'était pas nécessaire, lorsque l'un codait, que l'autre soit présent, à moins d'avoir besoin du recul d'un avis « extérieur ». Le travail ensemble consistait principalement en une première analyse des tâches et comment les aborder dans leur globalité, ainsi que la répartition. Une autre phase de travail ensemble était la mise en commun/fusion lorsque le code de l'un était nécessaire au code de l'autre.

Cette répartition du travail s'est faite assez naturellement. Sur le rendu 1 la lecture de fichier était la difficulté. Noam s'est occupé de poser les bases principales du programme (shape, tests d'erreur de base), et Talia s'est occupée de la lecture et des quelques tests d'erreurs qui necessitaient une bonne maîtrise de la méthode de lecture. Sur le rendu 2, GTKmm était la nouveauté, ainsi, pendant que Noam s'occupait de l'interface graphique, Talia a fait le reste. Enfin lors du rendu 3, Noam s'est occupé de la majeure partie des coraux et Talia du reste et des scavengers. Cette répartition a plutôt bien fonctionné et a été efficace.

Bugs :

Il n'y a pas de gros bug à proprement parler si ce n'est ce problème de suppression totale de mise à jour due à une mauvaise interprétation de la donnée. Nous n'avons pas pu le résoudre étant donné que nous nous en sommes rendus compte bien trop tard, et que cela aurait impliqué de profondément réviser la quasi-totalité de la logique derrière le comportement des coraux.

Un bug qui est très régulièrement revenu est la segmentation fault, probablement dû au fait que nos entités sont toutes stockées dans des vectors qui sont très fréquemment modifiés. Nous ne sommes non plus pas à l'abri de bugs indétectés (comportement qui n'est pas complètement absurde mais qui reste faux, pour cause de donnée incomprise).

Autoévaluation :

Nous finissons ce projet avec une simulation fonctionnelle d'un certain point de vue. En effet, tant que plusieurs problèmes n'arrivent pas à une simulation d'écart la simulation se déroule bien. Évidemment ce n'était pas ce qui était souhaité et nous aurions dû le remarquer et y réfléchir davantage.

Pour ce qui est de l'environnement de travail, les séances d'exercices nous ont bien aidés. Nous déplorons tout de même la donnée du projet qui nécessite d'être complétée avec les informations données en cours et sur EdStem. Passer des heures à éplucher les très nombreux threads de la plateforme dans l'éventualité d'avoir raté une information est très chronophage, et la centralisation des informations, chose essentielle pour un projet de cette ampleur, s'en est retrouvée particulièrement altérée.