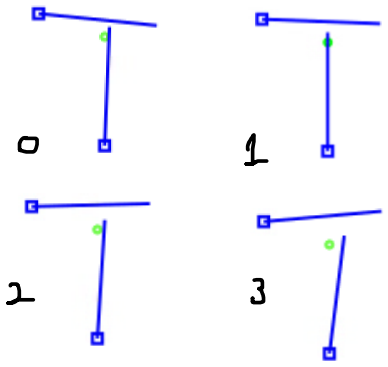
Projet Informatique – Sections Electricité et Microtechnique

Rapport du rendu 3

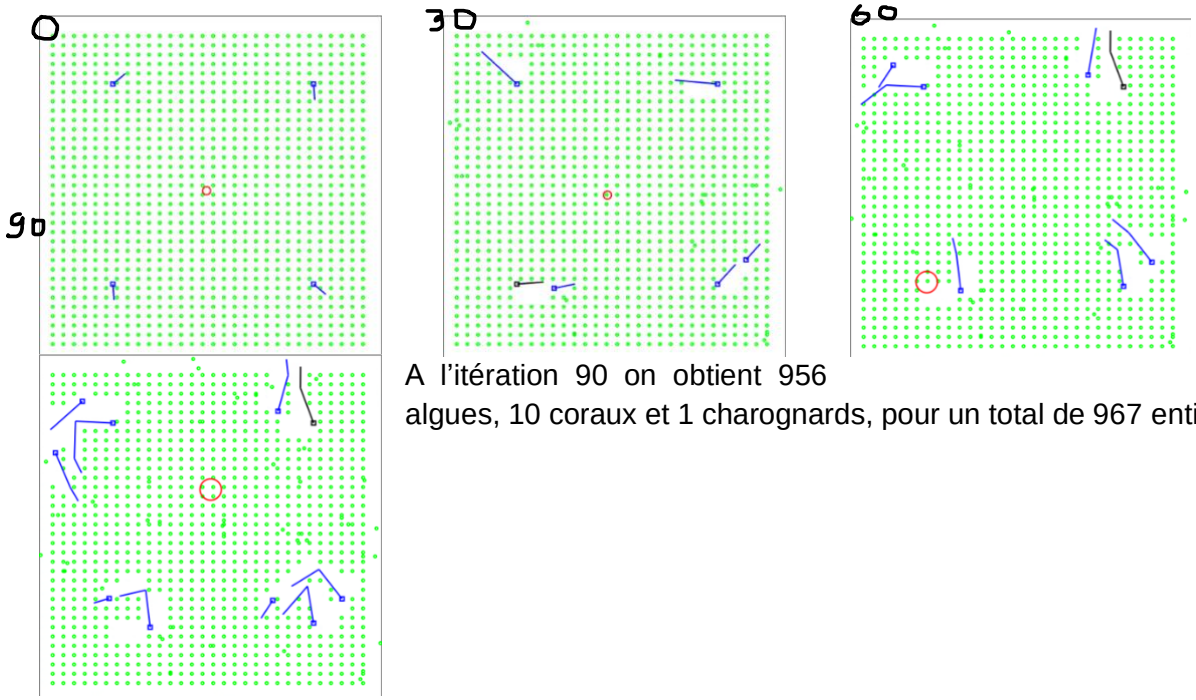
- Description de l'exécution :



Lors de l'exécution du test 43, nous appelons d'abords les méthodes de lecture de fichier qui vont lire le fichier et initialiser les entités. A chaque étape pour lesquelles c'est nécessaire, les méthodes appropriées détectant les erreurs sont appelées. Si aucune erreur n'est détectée, la fenêtre est créée avec les entités.

Puis, lors d'une mise à jour, la fonction step de gui est appelée et appelle la méthode step de simulation. Celle-ci va mettre à jour l'âge des algues et faire disparaître celle qui sont mortes. Ensuite, elle incrémente l'âge des coraux et essaye de les faire tourner ou de les faire manger. Si cela produit une collision on change la direction de rotation du corail. Dans l'exemple, les deux coraux veulent tourner dans le sens trigo. Cependant le corail du bas veut manger l'algue mais cela produirait une collision donc il ne la mange pas et il change de sens de rotation. Lors de la deuxième et troisième mise à jour, on appelle de nouveau step qui va faire tourner le corail du haut dans le sens trigo et inverse trigo pour l'autre.

Lors de l'exécution du test 46, la méthode step va également faire apparaître les nouvelles algues, mettre à jour les coraux, incrémenter l'âge du scavenger et vérifier qu'il ne soit pas mort, déplacer les scavengers vers les coraux morts et les faire manger.



A l'itération 90 on obtient 956 algues, 10 coraux et 1 charognards, pour un total de 967 entités.

- Méthodologie et conclusion :

Pour travailler sur ce projet nous avons utilisé VS Code et GitHub. Pour le mener à bien nous avons au début de chaque rendu partagé les tâches et ajusté de temps en temps lors de la conception pour s'organiser au mieux. Appart à ces moments-là où nous travaillions ensemble, une fois que les tâches étaient réparties nous étions indépendants et travaillions chacun de notre côté sur les différents modules. Pour éviter les *merges conflicts*, nous avions l'habitude de travailler chacun sur un module différent à la fois. Ce qui ferait à peu près 10% de travail en simultané et 90% en indépendant. Nous avons essayé de créer les modules du plus basique au plus complexe, pour toujours pouvoir les compiler et les tester.

Voici la répartition du travail pour chaque fichier source :

- Module graphic: 100% Daniel Roulin
- Module shape: 95% Daniel Roulin, 5% Joshua Hurlimann
- Module lifeform: 40% Daniel Roulin, 60% Joshua Hurlimann
- Module simulation: 40% Daniel Roulin, 60% Joshua Hurlimann
- Module gui: 50% Daniel Roulin, 50% Joshua Hurlimann
- Module projet: 50% Daniel Roulin, 50% Joshua Hurlimann

Pour créer un module, l'approche qui marche le mieux que nous avons trouvé est de le tester au fur et à mesure, via d'autre fichier en C++ ou en comparant les sorties. D'ailleurs, Daniel a également créé un script python pour automatiser les tests des deux premiers rendus. Un bug fréquent était de mettre à jour une copie d'une entité au lieu d'une référence. C'était difficile à trouver car tout semble marcher mais rien ne change. Enfin, nous avons eu quelques problèmes de synchronisation de version mais à part ça le projet s'est bien déroulé. Avec le recul, nous aurions peut-être dû travailler un peu plus côte à côte, ce qui aurait permis de gagner du temps. En effet, cela nous prenait parfois longtemps pour comprendre ce que l'autre avait fait. Pour conclure, je pense que nous avons plutôt bien réussi à recréer ce que le professeur voulait, même s'il doit y avoir des différences.