

### Organisation du Travail:

Notre équipe a utilisé une combinaison de **Git** et **VSCode** pour organiser et gérer notre travail. Voici comment nous avons procédé :

#### 1. Répartition des tâches:

- **Maud:** Interface utilisateur avec GUI et GTK.
- **Alice:** Modules principaux comme lifeform et simulation.
- Nous avons globalement bien réparti les tâches en fonction des compétences et des préférences de chacun.

#### 2. Collaboration:

- Nous avons principalement travaillé de manière indépendante, en nous envoyant nos travaux respectifs par mail ou via des branches spécifiques sur Git.
- Les tâches spécifiques étaient attribuées en fonction des forces de chaque membre.
- Nous avons effectué des réunions régulières pour synchroniser nos travaux et résoudre les problèmes ensemble.

### Proportion du Travail:

- **Travail Simultané en Groupe:** Environ 15%.
- **Travail Indépendant:** Environ 85%. Chacun travaillait de son côté sur des modules spécifiques l'une après l'autre.

### Réflexion Rétrospective:

Avec le recul, nous pensons que cette répartition était efficace pour notre équipe. Cependant, augmenter légèrement la proportion de travail simultané pourrait améliorer la synergie et réduire le temps de résolution des problèmes puisque parfois nous travaillions donc sur les mêmes problèmes simultanément.

### Le bug le plus fréquent et celui qui nous a posé le plus de problèmes:

La valeur de `default_random_engine` dépend de la machine utilisée, pour obtenir les résultats attendus, il faut faire les tests sur une VM et pas sur un Mac

Le bug le plus fréquent était le résultat « faux » renvoyé par la fonction « détection de superposition », c'est celui qui nous a fait perdre le plus de temps.

- En effet, les modifications exigées dans l'énoncé de la méthode `GeeksForGeeks` étaient très bien détaillées, on en a donc déduit que rien d'autre ne devait être modifié, alors qu'il fallait bien rajouter une condition supplémentaire lorsque 2 orientations étaient renvoyées comme étant colinéaires à cause de `epsil_zero` qui est non nul.
- Nous avons mis du temps à « accepter » de suivre l'énoncé à la lettre pour appliquer une méthode de superposition différente entre la lecture et la simulation : en effet, en simulation il est surprenant d'appliquer `epsil_zero` (0.5) pour la détection d'intersection entre segments, et d'autoriser à l'inverse que deux segments consécutifs se superposent exactement avec un écart angulaire inférieur ou égal à `epsil_zero`, et même inférieur ou égal à zéro (!), en valeur absolue.
- Pourtant, si on applique la même méthode de superposition que pour la lecture, le fichier `t29` réalisait 4 inversions de rotation, au lieu de 3, avant la mort.

### Brève auto-évaluation:

#### Auto-évaluation de notre travail:

##### Points Positifs:

- **Bonne répartition des tâches:** A la fin, chaque membre de l'équipe a eu des responsabilités claires, ce qui a permis une progression fluide du projet.
- **Utilisation efficace des outils:** nous avons réussi à installer `gtkmm` sur notre Mac et donc pour coder c'était plus simple que sur la vm.

- **Communication:** Les échanges réguliers via email et les réunions de synchronisation ont permis de maintenir une cohérence dans le développement.

#### **Points à Améliorer:**

- **Travail simultané:** Une plus grande proportion de travail collaboratif en temps réel. Comme nous travaillions pas ensemble, il y a eu quelques confusions et quiproquos.

#### **Auto évaluation de l'environnement de travail:**

##### **Point positifs:**

##### **Accès aux Assistants:**

- La présence d'assistants toujours disponibles pour répondre à nos questions a été un atout majeur. Ils ont pu clarifier des concepts complexes et résoudre rapidement des problèmes techniques.

##### **Utilisation de EdStem pour la Communication:**

- EdStem a servi de point central pour poser des questions et obtenir des réponses rapides, facilitant ainsi la résolution des problèmes en temps réel.

##### **Point négatifs:**

Je pense qu'on peut parler de la clarté de l'énoncé qui aurait pu être améliorée.

Nous aurions perdu moins de temps, en attendant les réponses qui ont mis du temps à émerger sur EdStem, si il avait été écrit dans l'énoncé des phrases comme :

- Lorsqu'il y a collision, la rotation est aussi annulée (et pas uniquement le fait de manger l'algue).
- L'extrémité du corail enfant est égale à l'ancienne extrémité de l'effecteur du parent.
- En mode EXTEND, l'effecteur du corail crée un nouveau segment en réduisant sa longueur (comme pour se « plier » lors de la prochaine rotation). L'extrémité du nouveau segment est égale à l'ancienne extrémité de l'effecteur.
- L'identifiant de la cible du scavenger n'est renseigné que lorsque le scavenger a atteint un corail mort disponible. Il reste vide tant que le scavenger est LIBRE.
- Les valeurs par défaut pour la création d'un nouveau corail sont INVTRIGO et EXTEND.

Il était difficile de faire confiance uniquement aux dessins de l'énoncé sans voir confirmé par écrit les différentes interprétations de ces dessins, que nous avons chacun compris différemment (nous étions d'abord partis dans une mauvaise direction en codant une extension et une reproduction qui génèrent un nouveau segment dont la base était égale à l'extrémité de l'effecteur précédent).

##### **- description de l'exécution**

##### **- T43.txt**

- Les coraux semblent s'éloigner de l'algue.
- Le corail du haut tourne dans le sens INVTRIGO tandis que celui du bas tourne dans le sens TRIGO



- **T46.txt**
- J'obtiens 963 algues, 10 coraux et 1 charognard

