Texte explicatif Création - Numérique :

UNIVERSITÉ DE NANTES

source : https://en.wikipedia.org/wiki/University_of_Nantes

L'objet de ce distanciel est d'implémenter un algorithme montrant les mouvements émergents de plusieurs agents autonomes.

Chaque agent se déplacera vers l'agent le plus proche d'un pourcentage de la distance les séparant. Si deux agents sont à une distance en dessous d'une limite (à fixer), il y a phagocytose : les deux agents sont remplacés par un seul agent dont le diamètre est tel que la somme des aires des deux agents donne l'aire de l'agent résultant.

Développeur :

- Coutand Bastien 485K : étudiant en deuxième année de licence informatique.

2019 / 2020

1 - Comment fonctionne le programme en général :

Au départ, il y a un total d'agent qui est égale à la constante *MAXAGENT* (ici 1000). Ils sont créés grâce à la procédure *initAgents(...)* qui instancie un agent avec une taille de départ prédéfinie grâce à la constante *STARTSIZE* (ici 10 pixels), et une couleur aléatoire sur les 16,581,375 couleurs possibles sous processing.

Ensuite, chaque agent recherche sont plus proche voisin (qu'importe la taille) grâce à la fonction *nearestIndex(...)*. Lorsqu'un agent a trouvé son plus proche voisin, il s'en rapproche grâce à la procédure *updateAgent(...)*, qui additionne des valeurs aux coordonnées de l'agent en question. Finalement, chaque agent va parcourir environ 50%, de la distance qui le sépare de son plus proche voisin.

Lorsqu'un agent entre en collision avec un autre agent, il y a phagocytose : si les coordonnées du centre de chaque agent se superpose à plus ou moins à la constante *LIMITFAGOCYTOSE* (ici 2 pixels, car ils ne superposent jamais vraiment aux pixels prêts à cause des erreurs liées au calcul arithmétiques) prêt, alors la procédure *phagocytose(...)* est appelé. Les deux agents sont remplacés par un seul agent dont le diamètre est tel que la somme des aires des deux agents donne l'aire de l'agent résultant.

Il y a une limite car, les agents ne vont pas pouvoir se superposer exactement sur une valeur proche, ce sont des flottant.

Toutes ces étapes sont modélisée grâce à la procédure *drawAgent(...)*, qui permet l'affichage des agents sur la fenêtre.

2 - Limite du programme :

On remarque, à partir de 5000 agents créés que le programme commence à atteindre ses limites, il met environ 2 secondes pour afficher la fenêtre. Plus le nombre d'agents augmente, plus le programme va mettre du temps à se lancer.

3 - Interactions possibles :

3.1 - Interaction souris :

S'il y a clic sur la souris, alors un nouvel agent est créé aux coordonnées de la souris, avec une taille de la taille de la liste, et une couleur aléatoire.

3.2 - Interaction clavier :

La touche clavier “**d**” permet d’activer la procédure de débogage *debug(...)* de l’algorithme qui va afficher en **noir** l’index de l’agent, en **violet** l’index de son plus proche voisin et une ligne reliant l’agent et son plus proche voisin.

La touche clavier “**e**” permet de stopper l'avancement des agents en désactivant la procédure *updateAllAgents(...)* qui permet le déplacement de tous les agents.

La touche clavier “**r**” permet de re-démarrer l’algorithme sans le relancer grâce à l’appel de la fonction *setup(...)*, qui initialise toute les variables.