LEVESQUE Paul-Antoine
VIGNE André
3i025

Compte-rendu de projet 3i025

L'objectif de ce projet était de faire se déplacer des personnages dans une map. Ces personnages ont le rôle d'agents. Ils doivent attendre un but précis, une fiole de couleur, et ne doivent pas entrer en collision dans leur parcours de la map. De plus, ils doivent à chaque fois emprunter le trajet le plus court possible.

La première chose que nous avons faite est l'écriture d'un algorithme A*, qui correspond à l'objectif du projet. Cet algorithme calcule, pour un personnage donné, son plus court chemin vers l'objectif en nombre de cases d'après la distance de Manhattan. Notre algorithme calcule cette distance et ajoute au fur et à mesure les cases à parcourir dans une liste propre au personnage. Ensuite, le personnage n'a plus qu'à parcourir successivement les cases de la liste et il arrive à son objectif. A ce stade, il n'y a aucune gestion des collisions et on observe régulièrement des personnages qui se croisent et se « rentrent dedans ».

Nous commençons donc à mettre en place les stratégies d'évitement des collisions. Dans le premier cas, nous choisissons la stratégie opportuniste. Pour chaque personnage, on calcule un A* depuis leur point de départ jusqu'au point d'arrivée. On les fait avancer, et lorsqu'un personnage tente d'avancer sur une case sur laquelle un autre personnage peut avancer, on recalcule un morceau d'A* à partir de la case précédente jusqu'au point d'arrivée. Ainsi, on ne recalcule pas la totalité du trajet et on économise du temps de calcul. Cependant, on peut observer par moment des personnages qui se croisent et se font face successivement à l'infini, ce qui fait qu'ils n'atteignent jamais leur point d'arrivée. Ces cas auraient été compliqués à gérer, mais nous avons choisi d'essayer quelque chose : on tire au hasard un nombre et selon une certaine probabilité, un des deux joueurs se met en attente pendant une itération, ce qui laisse le temps au joueur face à lui de calculer le morceau restant d'A* et de se déplacer. Cette méthode ne permet pas en revanche d'éviter les « sandwichs », cas également compliqué à gérer. Pour tester la validité de cette implémentation, nous avons grandement réduit l'écart de

disposition aléatoire des fioles et des personnages sur la map (10 cases, puis seulement 5 cases) pour vérifier si des collisions avaient lieu, ce qui n'était pas le cas.

Nous continuons avec une deuxième stratégie. Le but est d'identifier des chemins pour chaque personnage qui n'ont aucune case en commun. La première idée était de calculer en boucle des A* pour chaque personnage jusqu'à ce qu'ils n'aient aucune case en commun, et en ajoutant dans la liste des cases murées les cases qu'ils accumulent en commun. Cependant, même si cette solution peut paraître avantageuse, elle peut devenir compliquée à mettre en oeuvre sur certaines maps, et très couteuse en temps de calcul. On utilise une autre méthode : on calcule un premier A* pour le premier personnage, puis on ajoute tous les points par lesquels il passe dans la liste des cases murées. On réalise ces opérations pour le deuxième et troisième personnage, ce qui permet de ne calculer qu'un unique A* par personnage. Nous ne définissons pas de contraintes particulières pour l'ordre de passage. On aurait pu faire un tirage aléatoire, mais il aurait été judicieux de choisir en fonction de la taille de la liste des chemins proposés. On pourrait par exemple calculer la distance de Manhattan pour chaque joueur. On regarde quel joueur a la distance de Manhattan la plus longue, puis on calcule un A* pour ce joueur. Ensuite on prend les joueurs dans l'ordre décroissant de leur distance de Manhattan, puis on calcule un A* en prenant en compte les contraintes des joueurs précédents (ils ajoutent les cases par lesquelles ils passent dans la liste des cases murées). Ainsi, le joueur ayant le parcours le plus court passe en dernier et malgré toutes les contraintes ajoutées par les joueurs précédents, il aura peu de chances de dépasser le joueur qui avait le parcours le plus long. L'A* le plus long à calculer sera donc celui du joueur ayant la plus grande distance de Manhattan, et on économise encore du temps de calcul car on ne fait pas de calculs d'A* successifs pour chaque joueur.

Pour la troisième stratégie, nous ajoutons à l'A* une autre dimension. Cette dimension correspond aux positions d'un personnage à l'itération précédente et à l'itération suivante. On ajoute temporairement à la liste des cases murées ces deux positions, dans le but de prévenir des collisions éventuelles. Ici, on ne calcule qu'un seul A* par personnage au départ qu'on modifie au fur et à mesure lorsqu'on détecte une collision. Cela reprend en partie la première stratégie mais l'améliore car cela permet de gérer des cas de « sandwichs » ainsi que des face à face infinis entre deux personnages sans avoir besoin d'utiliser l'aléatoire et faire patienter les personnages durant une itération.