Récupération du champ visuel

En vue des simulations à venir, il est central de récupérer le champ visuel de chaque agent. Dans le dossier "Simulations", on trouvera deux scripts: "utils.py", contenant les fonctions centrales à la récupération de ce champ visuel, et "tests.py", dans lequel on effectue des tests pour vérifier le bon fonctionnement des dites fonctions.

1 Fonction get_others_in_visual_field

La fonction **get_others_in_visual_field** est la fonction principale. Étant donné le vecteur de positions des agents $\boldsymbol{X}$, l'ID i d'un agent ainsi que son orientation θ_i (un angle, par rapport à l'axe des abscisses), son angle de vision, la distance maximale à laquelle il voit et le rayon des agents, considérés comme des boules opaques, on retourne un vecteur contenant les ID des agents j vus par i. On représente en Figure 1 le résultat sur un exemple, l'agent i étant représenté en noir, les agents vus par i étant en orange (qui est ce que renvoie la fonction **get_others_in_visual_field**), les agents qui pourraient être vus par i mais qui sont cachés par d'autres agents en rouge (qui, avec les agents en orange, est ce que renvoie la fonction **is_in_visual_field**), et ceux hors du cône visuel de i en bleu. La distance maximale est prise à 30 et le rayon est pris à 1. La flèche verte représente l'orientation et les flèches grises caractérisent le cône de vision.

À noter que sur toutes les figures de cette note, il y a une déformation due à la différence d'échelle en x et en y.

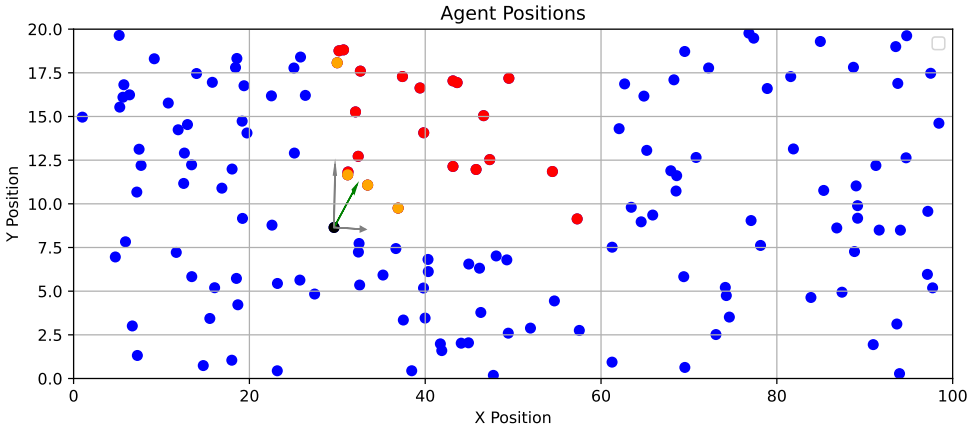

Figure 1. Exemple de relevé de champ visuel donné par la fonction **get_others_in_visual_field**.

2 Structure de la fonction get_others_in_visual_field et utils.py

Pour récupérer le champ visuel effectif via la fonction **get_others_in_visual_field**, on récupère d'abord la liste des agents qui sont dans le cône de vision de i, via la fonction **is_in_visual_field**, puis pour chaque agent de cette liste, on vérifie que cet agent n'est pas caché par un autre agent *de cette même liste* (en effet il ne peut être caché que par des agents de la même liste). On appellera pour faire cette vérification la fonction **blocks_target_from_ref**, qui regarde si un certain agent "inter" (pour intermédiaire) cache l'agent "target" de l'agent "ref" (pour référence). Toutes ces fonctions sont définies dans le script "utils.py".

Ce dernier contient aussi la fonction **is_in_contact**, qui est une redondance de **is_in_visual_field** avec pour seule différence les paramètres par défaut (angle du cône et distance maximale).

3 Script de tests

Le script “test.py” contient des tests pour les trois fonctions principales que sont **is_in_visual_field**, **blocks_target_from_ref** et **get_others_in_visual_field**. On donne en Figure 2 un résulat de la fonction **blocks_target_from_ref** : l’agent de référence “ref” est en noir, l’agent cible “target” est en vert, et l’agent “inter” (il pourrait y en avoir plusieurs mais on arrête la recherche dès que l’on en trouve un) qui cache “target” de “ref” est en rouge. Dans cet exemple, le rayon a été pris à 1. Notons que cette fonction n’a pas besoin de la donnée d’orientations, seulement des positions et du rayon des agents.

Figure 2. Exemple d’application de la fonction **blocks_target_from_ref**.

Enfin, on donne ci-dessous un autre exemple d’application de la fontion **get_others_in_visual_field** avec différentes valeurs de rayon : à gauche il est pris à 1, et à droite à 0.1.

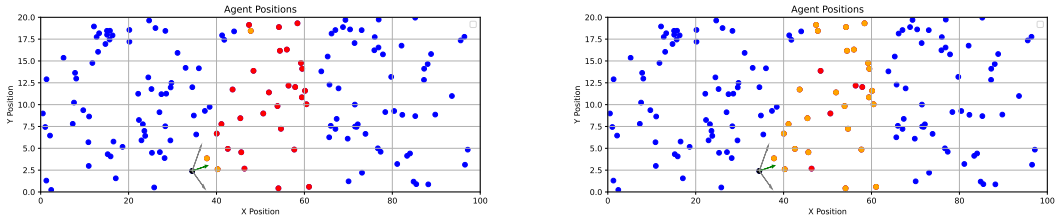

Figure 3. Exemple de relevé de champs visuel donnés par la fonction **get_others_in_visual_field**, avec à gauche un rayon de 1 et à droite un rayon de 0.1.