Mission de drones pour l'escorte de convois

Issam BANI
Olivier HOTEL
Stephen LARROQUE
Sous la supervision de : Mme Amal El Fallah Seghrouchni
Université Pierre-et-Marie-Curie, Paris 6, France

Table des matières

1 Introduction **1**
1.1 Scénario 1
1.2 Problématique 1

2 Modèle **1**
2.1 Environnement, physique et hypothèses . . 1
2.2 Agents 2
2.3 Convoi 2
2.4 Ennemis 3
2.5 Citoyens 3
2.6 Drones 3

3 Mode de « jeu » **6**

4 Conclusion et discussion **6**

1 Introduction

1.1 Scénario

Un convoi humanitaire composé de N voitures, dont une transportant une cargaison d'importance critique, doit atteindre un but en partant du hangar de la base. Pour cela, on suppose que le convoi connait la topologie du terrain, et qu'il peut donc calculer des plans de trajet.
Néanmoins, le reste de l'environnement est inconnu apriori, en particulier les ennemis : en effet, le convoi peut être attaqué. La mission est échouée si la voiture transportant la cargaison critique est capturée par un ennemi (la destruction des autres voitures n'influant pas sur la réussite de la mission).
Pour aider le convoi dans cette mission, des drones volants de type avion lui sont fourni. Ces drones doivent être autonomes et doivent pouvoir prendre des décisions seuls dans un environnement inconnu apriori (sans connaissance des autres agents présents sur le terrain) et dynamique (les autres agents peuvent bouger, des drones ou des voitures du convoi peuvent mourir, etc.).

1.2 Problématique

Il s'agira de créer une simulation modélisant ce scénario avec comme but de sécuriser le convoi, et d'implémenter un système de contrôle, de communication et de décision autonome et distribué pour les drones, en utilisant un paradigme SMA.
Il faudra donc se focaliser principalement sur l'implémentation des drones, les autres agents étant dans un cas réel des humains.
Nous supposerons donc les autres agents (normalement humains en cas réel) comme simulés : en d'autres termes, nous n'avons pas de contrôle direct sur ces autres agents.
Nous nous imposerons la contrainte supplémentaire d'éviter les interactions entre les autres agents (tels que le convoi) avec les drones. Par exemple, le convoi pourra communiquer sa position en wifi aux drones (ce qui est nécessaire pour que les drones sachent où se trouve le convoi) – ce qui peut être totalement automatisé dans un système réel en utilisant un système de transmission sans-fil automatique installé dans chaque voiture leader du convoi – mais en revanche les drones ne pourront pas communiquer avec le convoi. Cette supposition servira de contrainte pour l'autonomie des drones, afin qu'ils résolvent un maximum de problèmes par eux-mêmes sans aucune interaction, ce qui permet au convoi de ne pas se préoccuper des drones (et on imagine que dans un cas réel, un opérateur humain de convoi a bien d'autres soucis que de s'occuper des drones).
Néanmoins, les drones seront requis de contacter le QG par liaison satellite (ici appelée wimax) pour autorisation de tir lorsqu'un ennemi est détecté. La décision se fera donc principalement sur la détection et reconnaissance de dangers, mais pas sur la décision de tir qui sera donnée par un opérateur humain.

2 Modèle

Nous avons utilisé NetLogo3D version 5 avec l'extension BDI+FIPA par Sakellariou et al. [2].

2.1 Environnement, physique et hypothèses

Afin de réaliser une simulation réaliste de ce scénario, nous avons choisi de l'implémenter dans un environnement en 3D dynamique possédant un modèle physique qui contraint les mouvements.
Pour cela, nous avons fait les hypothèses suivantes :
– Dynamicité de l'environnement : un environnement est aléatoirement généré, ce qui permet de vérifier la robustesse et résilience des drones avec un environnement

changeant.
- Topologie du terrain et détection de collision avec obstacle : trois types d'obstacles sont générés : montagnes, lacs et rivières. Ce dernier type est particulièrement intéressant puisque des ponts sont générés aléatoirement sur les rivières, ce qui crée des points « bottleneck » où le convoi est obligé de passer. Il y a biensûr plusieurs routes possibles, mais ces routes sont toutes contraintes par les obstacles.
- Continu en X et Y : l'environnement n'est pas discrétisé dans ces dimensions (exception pour le calcul A* qui est discrétisé, mais des extensions existent pour le cas continu).
- Semi-discrétisé en Z : le changement d'altitude des drones est continu et progressif, mais des seuils d'altitude permettent de distinguer la basse altitude et la haute altitude (qui est l'altitude requise pour communiquer avec le QG en wimax). Il est à noter que la vitesse des drones est ralentie en haute altitude, et que ce changement est progressif et continu (si un drone est entre la basse altitude et la haute altitude, sa vitesse en sera ralentie proportionnellement).

Nous avons également implémenté un modèle physique et de ressources limitées :
- contrainte sur les mouvements et les vitesses de manière réaliste : un drone va un peu plus vite qu'une voiture du convoi, et ont un angle limité de rotation mais qui est plus important que celui des voitures. Les ennemis n'ont pas de limitation d'angle car ils sont à pied.
- ressources tels que le fuel et les munitions sont limités : si un drone ne rentre pas à temps pour se recharger à la base, il meurt quand il n'a plus de fuel. Les munitions sont également limitées pour les drones et les ennemis, néanmoins les drones peuvent les recharger à la base, les ennemis ne peuvent pas.

Enfin, diverses visualisations comme des codes couleurs ont été mis en place pour mieux distinguer les différents éléments et actions dans la simulation : par exemple, un leader du convoi sera en orange, alors qu'un suiveur en violet et la cargaison critique en jaune.

2.2 Agents

La simulation comporte quatre types d'agents avec quelques sous-types pour quelques-uns :
- Voitures du convoi, dont :
  - les leaders : elles calculent le plan de trajet pour atteindre le but (en utilisant A*). Au début de la simulation il n'y a qu'un leader.
  - les suiveurs : elles suivent la voiture devant elle (dont une qui suivra un leader).
  - la cargaison critique : dernière voiture du convoi, elle transporte la cargaison qui ne doit pas être capturée par les ennemis sous peine d'échouer la mission.
- Drones
- Ennemis, dont :
  - mobiles : ennemi humain à pieds qui peut se déplacer partout (sauf dans la base).
  - immobiles : ennemi tourelles qui ne peuvent se déplacer
    NB : ces deux types possèdent les mêmes possibilités de tir et de capture sur les drones et le convoi.
- Citoyens, dont :
  - agressifs : ce sont des citoyens qui peuvent attaquer les drones et le convoi, comme les ennemis.
  - neutres : ces citoyens n'attaquent pas, ils sont pacifiques.

FIGURE 1 – Les quatre agents de la simulation

La suite de ce rapport traitera en détail de la simulation des différents agents, et en particulier des drones.

2.3 Convoi

Le convoi démarre depuis la base sur une surface grise appelée « hangar ». La première voiture est choisie comme leader (en orange), et chaque autre voiture est liée à la voiture devant ou à coté d'elle, de façon à ce que chaque voiture suive en file indienne la voiture devant elle (suiveurs en violet). La dernière voiture contient la cargaison critique (en jaune).

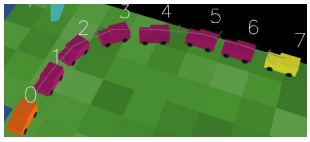

FIGURE 2 – Huits agents Convoi avec leader (orange), suiveurs (violet) et cargaison critique (jaune)

Une voiture leader, si elle ne possède pas encore de plan de trajet, va en calculer un en utilisant un algorithme WEIGHTED A*, qui est une généralisation de A* en ajoutant un terme multiplicatif à la fonction heuristique $h(n)$:

$$hw(n) = \varepsilon * h(n)$$

Ce qui donne les cas suivants selon la valeur de ε :
- ε=0 est équivalent à l'algorithme de Djikstra (= A* sans heuristique)
- ε=1 est équivalent à A*
- ε>1 est une approximation de A* généralement plus rapide avec une limite sur l'approximation d'au pire ε * coût du chemin A* optimal

La planification A* est une simulation d'un opérateur humain : dans un cas réel, un conducteur de convoi aura préparé un plan de trajet à l'avance en utilisant des cartes GPS et topologiques.

Mis à part la planification, le convoi est réactif, le convoi ne réagit donc que s'il est attaqué.
Par exemple, le convoi peut se scinder si une des voitures est détruite, on obtient alors deux convois :

- les voitures devant la voiture détruite continuent le chemin planifié
- les voitures derrière créent un nouveau convoi, et la voiture juste derrière celle qui a été détruite devient un leader. Elle calcule alors automatiquement un nouveau plan de trajet alternatif, en prenant des détours par rapport au chemin original. Cela permet de détourner l'attention des attaquants sur le premier convoi (qui continue sa route), permettant à ce deuxième convoi de s'échapper par une route alternative. Ce processus de calcul de plan est automatique dès lors qu'une voiture devient leader.

Dernier point : les leaders de convoi communiquent régulièrement par wifi leur position aux drones, et c'est la seule façon pour les drones de connaitre leurs positions.

2.4 Ennemis

Les ennemis (en rouge) sont soit de type mobile soit immobile (tourelle). Ils peuvent tirer des missiles sur les drones (en priorité) ou sur les voitures du convoi. Leurs munitions sont limitées, et leurs tirs ont une probabilité de rater la cible. Les ennemis ont également une visibilité limitée (une sphère autour d'eux d'une taille variable dans le GUI). Ils ont donc plus de mal à voir les drones en haute altitude que les drones en basse altitude, et voient encore plus facilement le convoi.
Enfin, leur but n'est pas de tuer la voiture cargaison, mais de la capturer : pour cela ils doivent arriver à l'atteindre sans être avant tué par les drones ou diverti par le scindage du convoi.
Les drones peuvent les détecter par une simple comparaison avec une banque d'image (Shape-based detection).

2.5 Citoyens

Les citoyens (en jaune) sont des agents un peu spéciaux rajouté spécifiquement dans le but de rajouter de l'incertitude dans la détection de dangers : en effet, les citoyens peuvent être agressifs (attaquant les drones et le convoi) ou pacifiques, et il n'y a aucun moyen apriori de savoir de quel genre ils sont.
Les drones doivent donc les détecter par comportement uniquement (Behavior-based detection).

2.6 Drones

Les drones sont le coeur de cette simulation. Ils sont implémenté comme des avions, et ont les contraintes physiques liées à ce type d'appareil (ils ne peuvent pas être stationnaires, ont une limite d'angle de rotation et ne peuvent se poser sur le terrain excepté sur la base pour se recharger).
Les drones sont des agents BDI, avec des capacités de communication de messages selon le protocole FIPA ACL.
Les drones ont également une vue limitée (une sphère les entourant) et de communication (également une sphère).

Gestion des ressources limitées. Les drones ont des ressources limitées, notamment le fuel (qui diminue constamment à chaque pas de temps) et les munitions (diminue à chaque tir sur un ennemi). Ils doivent donc régulièrement retourner se poser sur la base pour se recharger. La gestion des besoins est gérée ainsi :

- Fuel : on calcule le rapport suivant : distance manhattan entre la position actuelle et la base, divisé par la vitesse du drone. Ce rapport permet de savoir si le drone a encore assez de fuel pour retourner à la base. Quand (rapport - un petit terme de sécurité) est proche de 0, c'est qu'il est temps de rentrer à la base. Le terme de sécurité permet de s'assurer que le drone rentrera à la base avec une petite réserve de fuel pour parer à toute éventualité (sinon il rentrera quand le fuel permettra juste le déplacement sans prévoir les imprévu, comme par exemple rater l'entrée de la piste d'atterrissage).
- Munitions : lorsque le drone veut attaquer un ennemi mais qu'il n'a pas assez de munitions, alors il rentrera à la base avant de commencer la stratégie d'attaque.

La gestion des besoins prends le pas sur n'importe quelle autre intention : l'intention en cours sera continuée après le rechargement. En effet, les drones fonctionnent avec une architecture bi-couches : une couche pour les besoins primaires comme la gestion des besoins et l'évitement des ennemis ; une seconde couche pour les fonctions cognitives comme les communications ou la gestion des stratégies de patrouille, d'exploration ou d'attaque.

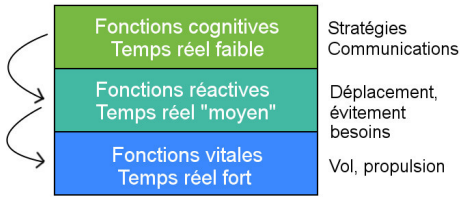

FIGURE 3 – Architecture modulaire que nous proposons pour une implémentation en cas réel. Dans la simulation, seules les couches « moyen » et faible sont implémentées.

Si un drone meurt (manque de fuel ou tué par un ennemi), il s'écrasera au sol et sera colorié en rouge.

Stratégies d'exploration/patrouille. Afin de protéger efficacement le convoi, les drones se répartissent des taches d'exploration et de patrouille afin de détecter et éliminer les dangers en amont. Nous avons implémenté plusieurs stratégies :

- Marche aléatoire : les drones explorent la carte de manière aléatoire pour détecter les dangers.
- Aller vers le convoi : permet de se diriger puis de rester à proximité du convoi.
- *Circular exploration* : permet de patrouiller en cercle autour du convoi afin de détecter les dangers à proximité.

FIGURE 4 – Patrouille circulaire

– *Path-Ahead exploration* : le drone parcourt le plan A* du trajet en amont du convoi pour détecter les dangers plus éloignés. Dans un cas réel, on suppose que le convoi pourrait communiquer un plan de trajet créé par des opérateurs humains utilisant des cartes GPS.

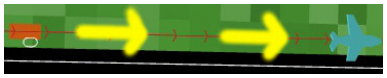

FIGURE 5 – Exploration en amont *Path-Ahead*

Il est à noter que ces stratégies ont une limite de temps : par exemple un drone qui fait un *Path-Ahead* ne va pas explorer tout le chemin à la fois (ce qui est inutile puisqu'un ennemi mobile peut apparaitre sur le chemin par la suite), mais plutot un bout de chemin, puis va revenir vers le convoi, et à la rigueur relancera ultérieurement une autre exploration *Path-Ahead* plus tard pour vérifier un peu plus du chemin et surtout si aucun nouveau danger n'est apparut. Le but étant d'accompagner le plus possible le convoi, et non de sécuriser tout l'environnement.
L'allocation distribuée de ces stratégies se fait de la manière suivante :
– Un drone est requis pour surveiller le convoi. Il doit donc toujours y avoir un drone à proximité du convoi, ne serait-ce que pour servir de relai de communication pour les autres.
– Les drones se communiquent leurs stratégies régulièrement.
– Les autres drones choisissent une stratégie aléatoirement et la communiquent aux autres.

Il est à noter que les drones ont des connaissances intrinsèques (leurs positions absolutes/GPS, leurs ressources comme le fuel et munitions) et des connaissances locales (les croyances, comme la position du convoi). Les drones utilisent donc uniquement des connaissances locales pour gérer leurs stratégies, ce qui fait qu'ils peuvent se perdre et ne pas retrouver le convoi suite à une mission qui les a mené trop loin. Ceci est tout à fait normal et fait partie intégrante du problème, ce que nous avons tenté de résoudre (au moins partiellement) avec le système de communication que nous détaillons plus bas.

Détection et Stratégies d'attaque. Tout d'abord, il est crucial que les drones évitent les attaques des ennemis. Pour cela, nous avons implémenté un évitement instinctif utilisant une modélisation du champ de vue des ennemis : lorsque les drones s'approchent d'un ennemi, la trajectoire de vol est déviée selon le champ de vue de l'ennemi. C'est donc un comportement réactif, de bas niveau, prenant la priorité sur toute autre action (à part la gestion des besoins qui est au même niveau). Ce comportement est assez similaire aux oiseaux ou aux insectes.
Lorsqu'un drone voit un nouvel individu, tout d'abord il consulte sa base de croyances (qui est aussi complétée par les autres drones) : si l'individu a déjà été détecté, le drone enclenche la stratégie d'attaque selon les résultats de la précédente détection. Sinon s'il n'est pas encore détecté, le drone va tenter une détection dans l'ordre suivant :
– *Shape-based* : reconnaissance par comparaison avec une banque d'images. Permet de reconnaitre les agents ennemis. Si c'est un citoyen, le drone va tenter le second type de détection.
– *Behavior-based* : reconnaissance par comportement en utilisant une heuristique multi-critères de comportement suspect (ex : emplacement embuscade, sur chemin du convoi, près d'une montagne, etc.). On fait une agrégation sur ces paramètres et si le résultat est au-dessus d'un seuil, alors l'individu est dangereux. Permet de reconnaitre les citoyens.

Au niveau de l'implémentation, ces détections sont très simples et ne sont que des simplifications de systèmes à développer plus précisément en cas réel. Par exemple la détection par image compare l'image d'un objet avec la banque d'objets de NetLogo en utilisant la fonction *shape*, mais dans un cas réel il faudrait comparer une image caméra avec une banque d'images en utilisant des critères comme GIST ou CENTRIST[3].
Lorsqu'un danger confirmé par détection est en vue, le drone va demander la confirmation de tir au QG (contrôlé par l'utilisateur), puis activer la stratégie d'attaque.
Dans le GUI, on peut choisir combien de drones sont nécessaires à une attaque :
– si un seul drone est nécessaire, le drone tire tout de suite sur l'ennemi (si biensûr le QG a donné le feu vert)
– sinon s'il est nécessaire d'être plusieurs, le drone va faire un Contract-Net : il va faire un appel d'offre pour trouver d'autres drones disponibles puis faire une attaque groupée.

Algorithme 1 Contract-Net pour tir groupé

```
1- Detecte si agressif. Si oui:
2- Question QG: autorisation tir?
    * Si oui et k = 1: tirer sur ennemi et stop.
    * Si k > 1: continuer à l'étape 3.
3- Aller vers le convoi (plus de chances de trouver d'
     autres drones) et, pendant le trajet, faire un
     Contract-Net:
  3bis- broadcast demander qui est disponible
  3ter- Autres drones répondent:
    * pas (si recharge ou surveille convoi)
    * oui sinon et se dirige vers l'appelant
4- Quand nb réponses = k-1, vol en cercle pour attendre
     regroupement avec autres drones
5- Quand regroupés, se diriger vers ennemi
6- Quand voit l'ennemi, tirer dessus
```

Note : les drones touchent l'ennemi à tous les coups (contrairement aux ennemis qui peuvent rater leurs cibles).

Note2 : les drones communiquent leurs détections et la réponse du QG aux autres agents.

Note3 : pour poser une question au QG, les drones doivent être en haute altitude, ou soit connecté en relai par un drone proche qui est en haute altitude. Ceci est automatiquement géré par les drones.

Communication. Les drones utilisent une architecture BDI, donc des connaissances locales pour leurs croyances et leurs stratégies. Afin de s'assurer qu'ils ne se perdent pas à la suite d'une mission, ou qu'ils ne redétectent pas deux fois un même ennemi, on doit s'assurer qu'ils ont des informations à jour.

Nous avons donc conçu un système de communication distribué, P2P, fonctionnant avec plusieurs mécanismes.

Il faut distinguer deux types de communications :

– WiFi : communication locale, à couverture limitée, de type broadcast aux voisins. Cela permet aux drones de communiquer entre eux et aussi au convoi leader de communiquer aux drones en WiFi.
– WiMax : communication globale (permettant d'atteindre n'importe quel individu), de type point-à-point. Cette communication est uniquement utilisée par les drones pour communiquer des questions au QG, et nécessite d'être en haute altitude.

Mécanisme automatique de communication P2P Le système de communication distribuée, basée sur le WiFi, a été conçu en utilisant les mécanismes suivants :

– Broadcast : un drone ou convoi communique une croyance à tous les agents drones voisins (dans la couverture wifi).

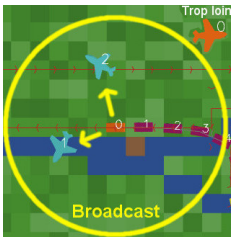

FIGURE 6 – Broadcast

– Relay : un drone relaye tout de suite à ses voisins toutes les informations qu'il est en train de recevoir (en filtrant les voisins qui ont déjà reçu le message, ceux qui sont dans la liste des destinataires du message original). Le drone ne relaye que les croyances pertinentes (ex : si une croyance n'est pas à jour, elle ne sera pas relayée. Cela évite qu'une croyance ne soit relayée en boucle indéfiniment).

FIGURE 7 – Relai de la position du convoi par le drone 0 vers le drone 2, les deux formant une chaine

– *Replay* (*Keep-me-posted*) : le mécanisme probablement le plus important de tous : il permet de *tenir au courant* un drone qui était parti en mission (ou perdu) et qui revient. Les autres drones vont alors lui rejouer toutes les informations qu'il a manqué (détections d'ennemis, position du convoi, etc.) depuis la dernière fois qu'ils ont été en communication avec ce drone. Cela permet aux drones de partager en permanence leurs informations, et de les recouper, offrant une certaine garantie que les informations de tous les drones soient à jour.

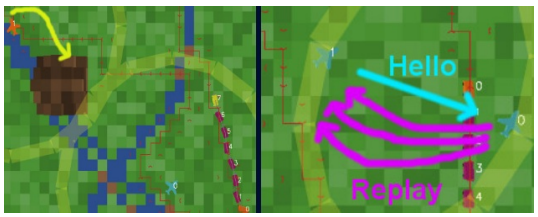

FIGURE 8 – *Replay* aka *Keep-me-posted* : le drone 0 rejoue toutes les informations manquées par le drone 1 (ennemis détectés, position du convoi, etc.)

Note : un code couleur permet de mettre en lumière les phénomènes de communication : en bleu quand un drone reçoit les communications du convoi (soit directement soit par relai d'une chaine de drones), en jaune quand un drone est connecté au QG (soit directement soit par relai), ou enfin en orange quand il n'est pas connecté ni au convoi ni au QG (mais il peut être connecté à des drones).
Ce système de communication permet de garantir que les croyances sont à jours pour chacun des drones, et ce de manière transparente. Le but est que les drones puissent utiliser leurs croyances et intentions BDI individuellement sans se soucier explicitement de la communication : en utilisant uniquement leurs connaissances locales, les drones utilisent en fait une base de données distribuée, globale entre tous les drones. Il n'y a pas besoin de requérir explicitement aux autres drones ce qu'ils font ou une croyance : tout est partagé automatiquement entre tous les drones.
Les drones sont donc une sorte d'intelligence collective, ou *Hive Mind*, dans le sens où chacun des drones partage les connaissances de tous les autres (ce n'est donc pas un *Swarm Intelligence* car l'intelligence n'est pas dans l'émergence des comportements mais dans le partage des connaissances).

Couverture limitée et Scattering problem La communication avec couverture limitée implique un problème connu appelé le *scattering problem* : comment faire pour que lorsqu'un drone part en mission, les autres ajustent leurs positions pour essayer de maintenir une chaine de communication entre tous les drones et le convoi ?

Pour tenter d'aborder ce problème, nous avons implémenté l'algorithme Connectivity-Preserving Scattering (CPS) par Potop-Butucaru et al. [1].

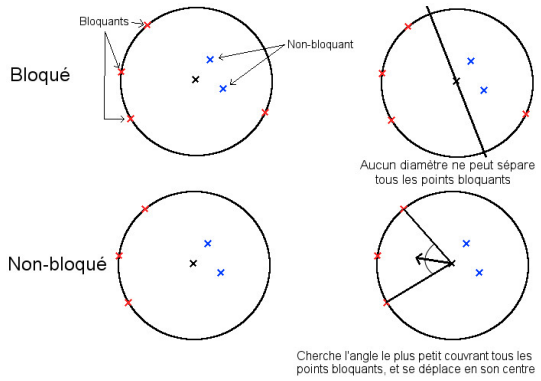

FIGURE 9 – Illustration des principaux éléments de l'algorithme CPS

Néanmoins nous avons dû adapter cet algorithme (et donc perdre les garanties de connectivité) car il suppose que les drones peuvent :

– être stationnaires
– tourner à des angles droits

En pratique, cet algorithme nous a permis de diminuer la taille de la couverture WiFi tout en conservant de bonnes performances de communication (pas trop de drones perdu). En cas réel, cela permettrait donc d'avoir des communications consommant moins d'énergie et plus discrets (car de plus petite portée).

3 Mode de « jeu »

Un mode de « jeu » a été implémenté pour proposer à l'utilisateur de prendre le contrôle du leader du convoi : le but est alors de conduire la voiture et le convoi jusqu'au but, sans que la cargaison ne soit capturée par un ennemi. Les drones aident le joueur de manière autonome, le suivant dans toutes ses actions. L'intérêt de ce mode est de pouvoir tester la résilience des drones avec un convoi ayant un comportement réellement humain (au lieu d'un comportement simulé avec A*).

Une intéraction supplémentaire entre le convoi et les drones a été ajoutée dans ce mode : le joueur peut demander à un drone d'aller explorer en avant de la direction où regarde actuellement le joueur. Cette nouvelle stratégie d'exploration remplace la stratégie d'exploration *Path-Ahead* (puisque dans ce mode il n'y a pas de plan A*).

4 Conclusion et discussion

Nous avons souhaité concevoir un modèle en 3D dynamique et relativement réaliste avec un modèle physique et des contraintes de mouvements et de ressources pour simuler des drones autonomes. Nous avons réussi à implémenter des systèmes de contrôle et de communication décentralisés efficaces ne demandant aucune interaction ni supervision par les humains (à part la communication automatique de la position/plan du convoi et l'autorisation de tir du QG), dans des drones robustes et résilients qui savent s'adapter face à des environnements et situations imprévus apriori.
Notre simulation en NetLogo passe à l'échelle assez bien puisque la complexité est en fonction du nombre d'agents cognitifs BDI (drones et ennemis) mais pas selon la taille de l'environnement (excepté pour le calcul A* du chemin mais il n'est fait qu'une fois au début de la simulation).
Nous pensons que notre modèle cognitif de drones et leurs stratégies sont transposables dans un cas réel. Il faudra biensûr réimplémenter ou étendre certains modules, en particulier :

– Mieux adapter le scattering (problème de recherche ouvert actuellement).
– Gestion de l'incertain (ne pas supposer que les positions sont certaines, les communications peuvent être perdues, etc.).
– Gestion des couches hardware en temps réel fort : il faudra réimplémenter les fonctions NetLogo pour gérer par exemple la propulsion des drones au lieu d'utiliser la fonction *forward*. Une piste possible est d'utiliser l'extension NetLogo GoGo.
– Localisation et Vision : il faudra implémenter des vrais capteurs, et donc des vrais algorithmes pour traiter ces informations. C'est là aussi un grand domaine de recherche ouvert et une grosse partie du travail sur un drone.
– Protéger les communications contre les attaques : il faudra améliorer la robustesse des communications (maintenance de cohérence et d'intégrité de BD ?) et les encrypter pour éviter l'interception et l'altération par les ennemis. Par exemple, le système actuel est fragile au *spoofing* : on peut envoyer des fausses informations dans les croyances des drones, et ceux-ci vont ensuite propager l'information partout sans la vérifier.

Bibliographie

[1] Maria Gradinariu Potop-Butucaru, Taisuke Izumi, and Sébastien Tixeuil. Connectivity-preserving scattering of mobile robots with limited visibility. In *Stabilization, Safety, and Security of Distributed Systems*, pages 319–331. Springer, 2010.

[2] Ilias Sakellariou, Petros Kefalas, and Ioanna Stamatopoulou. Enhancing netlogo to simulate bdi communicating agents. In *Artificial Intelligence : Theories, Models and Applications*, pages 263–275. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-87880-3. URL `http://users.uom.gr/~iliass/projects/NetLogo/`.

[3] Antonio Torralba and Aude Oliva. Modeling the shape of the scene : A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3) :145–175, 2001.