Université Pierre et Marie Curie

Rapport de projet
COordination et COncensus Multi-Agents

Drones Protect

Heraïz-Bekkis Daphné
Huam William
Année Scolaire : 2016 – 2017
M2 ANDROIDE

Encadrants :
Mme Amal El Fallah Seghrouchni
M. Cédric Herpson
Mme Aurélie Beynier

Table des matières

Partie 1

Introduction

Dans le cadre de l'enseignement universitaire COCOMA (*COordination et COncensus Multi-Agents*), nous avons réalisé un système de simulation 3D multi-agents en NetLogo.

1.1 Scénario

Été 2016, une nouvelle espèce apparaît sur la planète terre. Facile à capturer, les rangs de cette espèce sont rapidement décimés par les collectionneurs privés. Pour éviter la disparition de cette expèce atypique, l'ONU décide immédiatement de créer une réserve hautement protégée : la *POKESAFE*, et y transporte et libère les représentants les plus rares.
Dans le cadre de l'opération *FREEDOM4POKERAR*, un convoi composé de K véhicules de transports banalisés, dont l'un transportant un espèce rare, doit relier la réserve des POKERAR au départ du centre de naissance *INGRESS* le 5 décembre 2016, 11h00 UTC.
En tant que responsable de cette mission, nous disposons d'une connaissance *a priori* du terrain et sommes donc en mesure de planifier le trajet de notre convoi. Néanmoins, la position d'éventuels éléments hostiles et désireux de s'en emparer, par nature changeante, ne nous est pas connue.
La région où se trouve la réserve n'étant pas encore stabilisée, et la cargaison de haute importance, le commandement a affecté le nouvel escadron de drones autonomes de l'armée de l'air à la protection du convoie. Ces drones armés sont en mesure d'évoluer dans un environnement inconnu et ouvert, de réaliser tant des missions de reconnaissance que de destruction, de se coordonner tant entre eux qu'avec les unités au sol via leurs leader respectifs, et de prendre leurs propres décision en fonction des informations perçues.

1.2 Problématique

Notre objectif est de faire parvenir le véhicule transportant la cargaison à destination.
Dans ce but, nous avons du concevoir, modéliser et implémenter le système de communication et de décision dont est doté l'escadron de drones, en nous appuyant sur le paradigme agent. À partir du théâtre d'opération fourni, nous avons du réaliser une simulation du bon déroulement de cette mission.
Au cours de la simulation, nous supposerons les agents comme simulés, nous n'aurons donc aucun contrôle sur ces derniers.
Par soucis de réalisme, la simulation comportera plusieurs contraintes au niveau des agents et de l'environnement :

— Les drones composant l'escadron peuvent partager leurs informations sans contrainte de communication.

- Les drones ne peuvent échanger les informations dont ils disposent qu'avec les unités de même type à portée.
- Les unités ennemies (terrestres) sont dotées de comportements hostiles.
- Les différents véhicules du convoi sont dotés de comportements définis (calcul de chemin, évitement, dispersion).
- Les drones et unités ennemies sont limités par des resources finies (carburant, munitions) et doivent se ravitailler.
- Les drones peuvent agir selon divers comportement (exploration, protection, destruction, coordination).
- Les drones et le convoi adaptent leurs stratégies et rôles selon les pertes et le terrain.
- Centralisation de l'information chez les drones et mise à jour des connaissances.
- Communication drones-convoi par l'intermédiaire des leaders.
- Exploitation des données sur les changements de l'environnement.

La mise en oeuvre de ce projet a été réalisée sur la plate-forme de simulation NetLogo 3D. Nous avons aussi utilisé l'extension BDI de Sakellariou, Kefalas et Stamatopoulou [1].
Le code source du projet présenté ci-dessous peut être téléchargée à cette adresse `https://github.com/DaphneHB/Cocoma`

Partie 2

Modélisation

Dans l'idée de la réalisation de ce projet, nous avons du modéliser et faire interagir plusieurs entités, chacune bénéficiant de composantes qui lui sont propres.

2.1 Environnement

Le terrain de la simulation est une zone en trois dimensions (*x, y et z*) que nous avons définie arbitrairement. L'environnement dans lequel évolue les agents est cependant, lui, organisé aléatoirement et est dynamique. La génération aléatoire de cet environnement nous permet alors de tester objectivement la robustesse de notre implémentation. Cet environnement se veut réaliste et représente la nature de par sa topologie, comportant des zones, telles que des lacs, des rivières, des ponts, des montagnes, lieux inaccessible aux agents terrestres ou aériens. Les délimitations du monde n'étant pas fixées, c'est-à-dire qu'un agent pouvait se permettre de passer d'un côté à l'autre de la carte comme par 'téléportation' en arrivant aux limites de celle-ci, nous avons ajouté un obstacle théoriquement implicite : les murs du terrain sont ainsi considérés comme des obstacles lors des déplacements des agents, pour que ceux-ci restent restreints à des mouvements réalistes.
Pour la tenue de vol des drones, des seuils d'altitude ont été mis en place pour distinguer la basse altitude et la haute altitude.
Toujours dans la même optique de réalisation, des différences au niveau des vitesses respectives des agents sont à noter. En effet, un drone avance plus rapidement qu'une voiture du convoi tandis que cette dernière possèdera la même vitesse qu'un ennemi, car supposé lui aussi avoir le même moyen de locomotion. Enfin, les drones et ennemis possèdent des ressources limitées à l'utilisation. À force d'utilisation, le carburant ainsi que les munitions dans les deux camps sont réduits. Le ravitaillement à la base est cependant possible pour les drones, ce qui n'est pas le cas pour les ennemis. Nous verrons par la suite le fonctionnement du ravitaillement plus en détail.
En NetLogo, une case potentielle au déplacement des agents s'appelle un "*patch*", nous réutiliserons ce mot à l'avenir.
Au cours de notre simulation, et pour distinguer les différents états dans lesquels les agents se retrouvent, un code couleur a été utilisé.

Orange : Leader des drones (resp. convoi)

Jaune : Pour le convoi, représente le convoi à protéger. Pour les drones et ennemis, représente un agent en plein tir.

Rouge : Déplacement aléatoire de l'agent (se déclenche par défaut si aucun autre comportement ne se lance).

Violet : Suivi aléatoire d'une des voitures formant le convoi par les drones.

Bleu : Uniquement pour l'ennemi, lorsqu'un drone ou convoi est dans son champ de vision mais pas assez proche pour le tir. Implique le rapprochement de l'unité.

La simulation comporte quatre types d'agents.
Ennemi : Unités terrestres mobiles et hostiles au convoi.
Convoi : Unités terrestres à escorter et protéger.
Drone : Unités aériennes de protection du convoi.
Bullet : Balles tirées par les ennemis et les drones, continuent tout droit même si leur cible bouge et blesse tous les agents équitablement, sans distinction.
Ces agents ont en commun de tous posséder des jauges de vie se décrémentant à chaque blessure, pouvant donner lieu à leur mort si celle-ci est nulle.
Les agents sont aussi équitablement limités pour leurs déplacements et leur vision : un agent ne peut ni traverser une montagne, ni marcher sur l'eau (ni même nager), ni voir à travers une montagne ou de l'autre côté du terrain.
Les couleurs que nous attribuons aux agents sont des repères pour la/les personne(s) qui lancera/ront cette simulation et ne sont, jamais durant la simulation, utilisées par un autre agent.
Concernant les croyances (ou *beliefs*), chaque agent à ses propres croyances. Chaque type d'agent à une croyance de type commun "*list-beliefs*" comportant la liste des croyances, dynamique, que ce type d'agent gère, où chaque valeur d'une croyance de type "*list-beliefs*" est un type, tel que "*ennemies-at*", croyance pour stocker les positions des ennemis.

2.2 Ennemis

Il s'agit des unités terrestres hostiles au convoi. Celles-ci sont mobiles, elles peuvent se déplacer dans le terrain en respectant les contraintes d'obstacles. Ils sont, par défaut, de couleur rouge et se déplacent aléatoirement dans le terrain et de façon très basique, à chaque itération (tick), l'ennemi effectue une rotation aléatoire puis essaie d'avancer de *e-speed* patch - où e-speed est sa vitesse, choisie par l'utilisateur au lancement de la simulation et dépendant de la *simu-speed* -, il peut par conséquent être bloqué dans son mouvement par les différents obstacles qu'il rencontre. Sa situation se débloquera donc par la suite lorsqu'il ré effectuera une rotation.
Un ennemi a un champ de vision *e-vision* lui permettant de voir à 360 °. Lorsque ce dernier à détecter une potentielle cible, telle un drone ou un véhicule du convoi, il change alors de couleur et vire au bleu, se dirigeant vers l'entité pour réduire la distance le séparant d'elle afin d'être à portée de tir. En effet, pour pouvoir attaquer un autre agent, d'un autre type que lui-même, l'ennemi doit être positionné dans un rayon de *e-dist-tir*. Une fois se périmètre atteint, l'ennemi passe au jaune, s'arrête et attaque immédiatement sa cible. Par ailleurs, nous avons décidé arbitrairement que les ennemis se focalisaient en priorité sur les drones à portée plutôt que sur le convoi, simulant ainsi un échange de tir. Les ennemis tirent des balles à une fréquence spécifique : *e-frequence-tir* * *e-dist-tir* / *simu-speed* faisant alord perdre à la cible un point de vie à chaque balle reçue.
Pour une équité totale, les ennemis ont eux aussi une jauge de vie *e-life* diminuant à jauqe tir reçu, ennemi ou allié. Effectivement, une balle "perdue" qui se voit toucher un agent ne fait aucune distinction et blesse l'agent qu'elle atteint, qui que soit son envoyeur.
Pour les ennemis, la base de décollage des drones et le hangar du convoi sont des obstacles, ils ne peuvent s'y déplacer, ni ne peuvent voir ce qui s'y passe. Nous avons aussi délimité une zone "safe" où les ennemis ne peuvent apparaître, par soucis de lisibilité pour le début de nos simulations, évitant ainsi des combats dès les premiers instants de notre simulation

2.3 Convoi

Le convoi démarre du coin en bas à gauche du terrain : le hangar, à proximité de la zone de décollage des drones.
Le convoi est composé de voitures, reliées par des liens, en file indienne. Chaque convoi possède un leader, colorée en orange pour la distinguer. Au cours de la simulation, un seul véhicule sera le plus critique, celle-ci sera en jaune pour attirer facilement notre attention au cours de la simulation, et sera la dernière voiture du convoi initial. Les voitures placées entre le véhicule leader et le véhicule critique du début de partie sont des suiveurs et sont de couleur violette tout au long de la simulation. Par défaut, la première voiture placée dans le hangar est la voiture leader. Au cours de la simulation, plusieurs véhicules leaders de convoi peuvent émerger mais il n'y a toujours qu'un véhicule leader par convoi, ce qui implique que le convoi initial peut se retrouver scindé en deux (ou plus). Cependant, même lors d'une scission ou de morts, le véhicule à cargaison critique reste toujours le même. Dans notre cas, la scission se fait par décision du prochain leader additionnel : lorsque ce dernier atteint un certain seuil en terme de point de vie, il décide de "couper les ponts" avec le véhicule immédiatement devant lui et s'auto-proclame leader. Cependant, le précédent leader, s'il n'est pas mort entre temps, reste aussi leader mais de son propre convoi. Chaque leader s'occupe alors des mises à jour du trajet global et des communications avec les drones.
Le but du convoi est, en partance du hangar, point de départ, d'arriver à sa destination finale, son objectif, avec pour priorité de ramener le véhicule critique à cet endroit précis.
Les convois n'ont aucune capacité de tir et ne peuvent se défendre contre les attaques ennemies. Malgré cela, il peuvent tout de même perdre des points vie dans le cas où ils se font toucher. Dans l'implémentation faite, chaque véhicule, nommé convoi, est autonome, possède ses propres point de vies *c-life* et ses propres croyances et décisions. Malgré cela, tous les véhicules doivent arriver au point objectif.
Au démarrage de la simulation, les véhicules ont une connaissances totales des reliefs et obstacles de la carte et peuvent ainsi calculer le chemin le plus rapide pour atteindre leur objectif en passant par les chemins praticables existants. Ce calcul est réalisé par le convoi leader au tout début pour que ce dernier mène sa suite sur le bon chemin.

En ce qui concerne le déplacement des convois, nous avons préféré ne pas garder la méthode de *convois-think* donnée en cours. Nous avons souhaité doter tous les véhicules des convois d'un chemin pour éviter que les voitures non leader ne suivent bêtement les convois de devant au risque de traverser impunément une montagne ou une rivière. En outre, nous avons considéré plus pertinent que chaque voiture ait le chemin en tant que variable propre à elle et permettre ainsi plus de flexibilité dans le cas d'ajout de nouvelle fonctionnalité. Bien entendu, nous ne recalculons pas un chemin complet pour chaque voiture.
L'astuce est de demander le vrai calcul du chemin au leader, et d'effectuer pour les autres, seulement le calcul du chemin entre la voiture précédente et elle même pour enfin concaténer les chemins obtenus. La difficulté de cette solution a été de lier la partie agents et la partie globale, la fonction *plan-astar* ne pouvant être appelé lorsque l'on s'occupe des convois.

Ce calcul d'itinéraire, très efficace, ne prend cependant pas en compte les positions des ennemis qui changent régulièrement. En effet, les unités terrestres hostiles étant mobiles, il faut régulièrement mettre à jour ses connaissances/croyances du monde : les *beliefs*. Cependant, il s'avère être très lourd de re-vérifier à toutes les itérations (*ticks*) la présence ou non d'ennemis dans le champ de vision *c-vision* d'un véhicule. Nous réalisons donc cette vérification à intervalle de ticks régulier. À ce moment, les précédentes croyances concernant les positions

ennemies sont oubliées, pour ne pas surcharger la carte interne et locale à chaque véhicule, puis immédiatement après re-vérifiées en observant le monde alentour et sauver jusqu'à la prochaine vérification. Seules les coordonnées des patches contenant des ennemis sont sauvegardées. Ce calcul régulier et réalisé par chaque véhicule de la carte. Après avoir mis à jour ses croyances se concernant et si le véhicule n'est pas un leader, alors celui-ci envoie ses croyances accumulées à son leader de convoi. À la fin d'une itération, le leader du convoi récupère alors tous les messages reçus de ses suiveurs concernant les localisations ennemies et les ajoute à ses propres croyances. Cela réalisé, le leader place alors momentanément les patches comme étant des zones ennemies : *set zone-ennemies true* ainsi que les *c-vision*-1 patches dans le rayon de l'ennemi pour la marge d'erreur d'attente de re-vérification. À partir de là, le leader vérifie s'il est nécessaire de recalculer le chemin à suivre jusqu'à l'objectif final, c'est-à-dire d'éviter les zones trop "*à risque*" car contenant des ennemis ou trop proche de leur présence. Après cette rapide vérification, les patches sont replacés comme normaux pour ne pas rendre moins réalistes les calculs et considérations des zones ennemies pour les autres leaders de convoi, bien que toujours considérés par ce leader comme étant les zones ennemies à son convoi.
Ce comportement simple de considération des ennemis et de recalcul du chemin, permet au convoi de s'adapter assez bien aux situations critiques et d'éviter de se retrouver en mauvaise posture. Cela est d'une grande aide pour alléger la protection des drones apportée aux convois.

Concernant la communication, les véhicules du convois ne peuvent communiquer que avec le leader de leur convoi, celui avec lequel ils sont liés, et ne peuvent lui communiquer que des croyances, *beliefs*, que lui-même connaît.

2.4 Drones

Les drones, de même que les convois, sont positionnés en sud ouest de la carte lors du début de la simulation. Nous avons lors de ce projet, doté les drones d'un BDI.
Ainsi au commencement, nous avons ajouté les comportements de marche aléatoire et de suivi de convois à la pile d'éxécution, puis le comportement de décollage des drones. Ajouté à cela, nous avons demandé aux drones d'ajouter les comportements offensifs à intervals de temps réguliers : les drones essaient par exemple de trouver les ennemis les plus proches pour les détruire tous les 50 ticks. Néanmoins, ils ne s'acharneront pas à se diriger vers des ennemis proches car 10 ticks plus tard ou si le convoi s'éloigne trop, ils retourneront à leur comportement de suivi de ce dernier.
De même, le controle du carburant a été effectué sur le même modèle, un drone parmi ceux ayant le moins de carburant est désigné, ce dernier ajoute donc la suite de comportements liés au retour à la base et au ravitaillement en carburant à sa pile. Il faut noté que chaque drone ira néanmoins se ravitailler s'il atteint un seuil critique de carburant.
Ce système permet un roulement des drones présents à la surveillance du convoi. Nous avons par ailleurs été confronté au problème de "comment le drone rejoint le convoi ?". Pour résoudre cela, le drone quittant le groupe, regarde le chemin du leader du convoi, et fait une estimation de l'endroit où la voiture se situera le temps que le drone fasse l'aller retour. Ce moyen permet en général, le bon retour du drone, cette solution ne s'appliquant pas au cas où le convoi change de direction entre temps. Dans ce dernier cas, le drone sera perdu et effectuera des mouvements aléatoires, dans l'espoir de retrouver son groupe ou un convoi.

En ce qui concerne les communications des drones, nous avons travailler sur la réception de certains messages envoyés par les convois comme la liste des membres constituant un convoi.

Avec cette liste reçue, le leader des drones met à jour sa croyance concernant les différents convois existant. Par exemple, si le leader a connaissance d'un seul convoi constitué de voitures numérotées de 0 à 9, s'il reçoit un message lui faisant part de l'existence d'un convoi de voitures numérotées de 6 à 9, il mettra à jour ses connaissances en faisant la fusion des données, en obtenant ainsi deux listes "0 à 5" et "6 à 9". Le leader devine ainsi en recevant une nouvelle liste dont les membres sont présents dans une des listes qu'il a en sa possession, qu'une scission a eu lieu, il propage ensuite ses croyances à ses coéquipiers.
Cette liste de convois sera ensuite utilisée pour coordonner les différents drones et les assigner à chacun d'eux.

De la même façon que les convois, les drones amassent localement à chacun des croyances sur les emplacements ennemis et les partagent avec leur leader drone. Ce dernier va alors mettre toutes ces informations en commun pour former sa base de croyance mais ne va rien en faire de plus en application avec le terrain. Le leader drone, et uniquement lui, va alors partager cette croyance avec le leader du convoi auquel il est assigné. C'est donc le leader convoi qui, comme pour avec les convois, va cumuler ses connaissances et celles reçues pour les prendre en compte dans le calcul du chemin le "*moins risqué*".

Partie 3

Conclusion

3.1 Synthèse

Au cours de ce projet, nous nous sommes familiarisés avec le langage de programmation NetLogo et au formalisme du BDI.
Nous avons visé de construire un modèle 3D autonome, adaptatif, dynamique et robuste en y incluant des contraintes de resources pour plus de réalisme. Dans ce but, après avoir pris connaissance du code, nous nous sommes efforcés d'optimiser certains aspects comme l'exploitation des connaissances des convois et des drones, certaines méthodes nous paraissant gourmandes en ressources, ou encore le déplacement des convois. Nous avons réussi à implémenter des systèmes de communication et de contrôle des agents décentralisés et plus ou moins efficaces, ne demandant jamais aucune interaction ou supervision humaine face à des environnements, reliefs et situations à priori imprévus. Il en ressort par ailleurs que seuls le nombre d'agents augemente les calculs et donc les temps de réaction, et que la taille de l'environnement n'est pénalisante que pour le calcul du A* .

3.2 Améliorations

Nous nous sommes penchés durant ce projet sur certaines améliorations possibles comme le calcul du chemin par l'algorithme A* : actuellement, celui-ci exclu les zones où le convoi et les drones observent des ennemis. L'idée serait d'y ajouter un poids pour que le convoi puisse opter pour un chemin possédant des ennemis en faible nombre au lieu de faire un détour pouvant être conséquent. Les paramétrages des différents poids seraient bien entendu à faire.
Les autres pistes que nous avons explorées sont surtout une optimisation du code afin de le rendre le plus générique possible et réutilisable pour tout les types d'agents, notamment en ce qui concerne les beliefs, les réceptions de messages et leurs traitements (fusion des données et suppression des croyances redondantes).
Enfin, se pencher sur les stratégies connues en matière de protection et d'escorte de convois afin d'améliorer la priorisation des intentions de nos drones et ainsi améliorer leur perspicacité.

Bibliographie

[1] Ilias Sakellariou, Petros Kefalas, and Ioanna Stamatopolou. Enhancing netlogo to simulate bdi communicating agents. *Artificial Intelligence : Theories, Models and Applications*, pages 263–275, 2008.