الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – Guelma -

Faculté des Mathématiques, d'Informatique et des Sciences de la matière

Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique

Option : Systèmes et technologie de l'information et de la communication

Thème :

Coordination bio-inspirée d'un essaim de robots : application au problème de recherche et de sauvetage

Encadré Par :

Dr. Ouarda ZEDADRA

Présenté par :

Nor El Houda MAIZI

Septembre 2020

Dédicace

I would like to dedicate this work:

For each person who gave me a positive energy, who helped me go through these five years

My Grand-Mother the light of my life

My mother, father and of course my brother Amine

All my cousins and Uncles especially the sweet Balkis Nada Elrihane, Yassmine and Mofida – well I would say Razane Sajida her daughter and Yussuf were the most positive thing that ever happened to me in this five years.

Friends who come and went that were a big impact both positive and negative on me such as Salah Eddine, Imen, Hamida and the list goes on.

My People and other family in CreatiVa club who; which I am not exaggerating changed my life; each member of us the newest and the oldest.

And all the people that I actually known in my life such us Yasser Benyounes, Ziou Abdrezzag, Benzaid Aymen, Fadlia Bessma, Djakdjak Hadjer, Chemmakh Amira …

The teachers at "Guelma Informatique Department" my role models

a special gratitude to the beautiful Bouthaina who was just there whenever

and Myself for being able to take all these struggles and survive

Maya

Remerciement

I want to give my gratitude to all those in whom, by their presence, their support, their availability and their advice, I had the courage to accomplish this work.

Above all, it seems opportune to give thanks to Allah for giving us the will and the patience and the ability to accomplish this work.

i will start by thanking myself for being able to complete this thesis and all these years, it was so pleasing to be able to learn all this knowledge.

I would love to be able to thank my parents for the unconditional support they have shown. Thank you for the financial, moral, psychological and material support. If I am here today, it is thanks to you!

I would like to thank the members of the jury for having done me the honor of being part of this work.

I would like also to thank my supervisor Dr. ZEDADRA Ouarda, teacher at the University of GUELMA, for all the help, guidance, as well as for her precious advice while the realization of my thesis. I learned a lot from her and it was a great pleasure to work with her.

I would also like to thank all the people who, near or far, friends or family for their support, encouragement and help in the preparation of this brief.

Finally, I will thank my friends and colleague of the class of 2020; and of course our dearest computer science department : each member whether it is a students or a teacher or just a worker there for all these years we spent together, in the best moments as in the worst even thought there were not any bad moments.

Résumé

L'application de recherche et de sauvetage consiste à naviguer dans un espace catastrophique selon une stratégie de déploiement dans le but de chercher et transporter des victimes à certaines localisations (dépôt). C'est une tâche qui se compose de deux comportements de base : l'exploration et le retour au dépôt. Ces deux comportements constituent des défis dans la littérature et plusieurs travaux veille à leur résolution.

Nous proposons dans ce travail d'utiliser un essaim de robots pour la résolution d'une mission de recherche et de sauvetage. En se concentrant, d'un côté, sur la dispersion de robots qui assure une large couverture de l'environnement et donc une grande éventuelle de localiser plus de victimes, et de l'autre côté, sur le retour rapide et non couteux au dépôt. Les systèmes biologiques forment de meilleures inspirations dans de telles conditions. L'exploration est ainsi assurée par la marche aléatoire Lévy (Lévy walk) et l'algorithme Dragonfly. Tandis que, le retour est assuré par des spots lumineux qui guident le robot jusqu'au dépôt. Le robot va suivre la lumière jusqu'au dépôt et n'as pas besoin de mémoriser sa position.

L'algorithme a été implémenté sous la plateforme de robotique mobile ARGoS. Plusieurs simulations ont été réalisé sous la même plateforme et une analyse des résultats montre l'efficacité de notre proposition par rapport à la marche aléatoire.

Mots clés : intelligence en essaim, robotique en essaim, exploration Multi-Robots, recherche et sauvetage Multi-Robots, Algorithme Levy Walk (LW), algorithme Dragonfly.

Sommaire

Liste des figures

Chapitre 1 :

Chapitre 2 :

Chapitre 3 :

Chapitre 4 :

Liste des tableaux

Chapitre 2 :

Chapitre 4 :

Liste des équations

Chapitre 3 :

Introduction générale

L'intelligence en essaim, est un champ de recherche scientifique dérivant de l'intelligence artificielle distribuée, qui s'intéresse aux processus distribués d'organisation et de résolution de problèmes présents dans un certain nombre de sociétés animales et dans des systèmes artificiels qui en sont inspirés. Un essaim se composent de plusieurs individus dotés de capacités de calcul, de mémorisation et de communication limitées. Ces individus agissent selon des règles locales simples résultantes des informations qu'ils perçoivent au local mais qui produisent des phénomènes cohérents à l'échelle global. L'interaction entre actions individuelles simples peut de façons variées permettre l'émergence de formes, organisations, ou comportements collectifs, complexes et cohérents, tandis que les individus eux se comportent à leur échelle indépendamment de toute règle globale. Cette branche de recherche forme une source d'inspiration pour la conception de plusieurs systèmes artificiels comme le routage de marchandise, *la robotique en essaim*…etc.

La robotique en essaim c'est une branche de la robotique mobile qui prend ses inspirations de l'intelligence en essaim. Elle forme un cadre applicatif de l'intelligence en essaim. Elle permet de coordonner l'activité de plusieurs robots afin qu'ils accomplissent collectivement une tâche. Plusieurs applications de la robotique en essaim existent : le mouvement coordonné, le nettoyage, le foraging, la recherche et le sauvetage…etc.

Le problème de recherche et de sauvetage constitue une instance du problème de foraging dans le monde réel. Elle consiste à naviguer dans l'espace selon une stratégie de déploiement dans le but de chercher et transporter des victimes a certaines localisations. C'est une tâche qui se compose de deux comportements de base : l'exploration et le retour au dépôt. Ces deux comportements constituent des défis dans la littérature et plusieurs travaux veille à leur résolution. Une exploration est plus efficace s'il y a eu une détection d'un grand nombre de victimes dans des temps réduits. La dispersion des robots sur des zones distinctes peut assurer une exploration efficace où la recherche sera plus rapide, le nombre de victimes collectés sera plus élevé et les collisions seront évitées. Aussi, réduire le temps de retour va réduire le temps total de recherche et sauvetage.

Nous proposons dans ce mémoire une contribution pour résoudre efficacement le problème de recherche et sauvetage en utilisant un essaim de robots. Nous utilisons : (1) dans la phase d'exploration la marche aléatoire Dragonfly Lévy connue par sa force à disperser les robots dans la totalité de l'environnement ; (2) dans la phase de retour au dépôt, des spots lumineux qui guident les robots vers le dépôt sans besoin d'avoir une mémoire pour sauvegarder des informations complémentaires (comme la position du dépôt...). La proposition a été implémenté sous la plateforme ARGoS et les simulations montrent de bons résultats en comparaison avec la marche aléatoire simple.

Ce manuscrit se trouve partagé en 4 chapitres :

- **Le premier chapitre** présente de manière brève les définitions et les différents concepts reliées à l'intelligence et la robotique en essaim ;
- **Le deuxième chapitre** présente une synthèse des travaux reliés au problématique de recherche et de sauvetage avec un tableau de comparaison des travaux ;
- **Le troisième chapitre** décrit en détail la proposition faite et les algorithmes reliés ;
- **Le quatrième chapitre** présente l'implémentation, les simulations réalisées, les résultats obtenus et les discussions.

Nous terminerons ce manuscrit par une conclusion et quelques perspectives.

Chapitre 1 : Intelligence et robotique en essaim

I Introduction

L'intelligence en essaim est un sous-domaine de l'informatique naturelle et s'inspire du comportement collectif des insectes et autres animaux sociaux pour concevoir des algorithmes de résolution de problèmes (Li & Clerc, 2019)**.** Elle est caractérisée par une collection structurée d'individus aux capacités individuelles limitées interagissant et présentant un comportement intelligent collectif capable de résoudre des problèmes complexes **(Cruz et al.,2019)**. Les recherches sur l'intelligence en essaim sont principalement axées sur les applications permettant de résoudre des problèmes complexes, la conception de nouvelles approches et l'amélioration de celles existantes **(Bahel et al., 2020).**

La robotique en essaim est la nouvelle approche de la coordination de systèmes multi-robots, qui prends ses inspirations de l'intelligence en essaim.

Nous divisons ce chapitre en deux parties : dans la première, nous présentons quelques définitions de l'intelligence en essaim, ses propriétés, ses stratégies d'interaction et de transfert d'informations avec ses fonctions et applications, et enfin quelques algorithmes issus de l'intelligence en essaim. Dans la deuxième partie, nous présentons la robotique en essaim comme une branche de recherche inspirée de l''intelligence en essaim. Nous présentons quelques définissons, ses caractéristiques et ses applications, et enfin nous montrons les plateformes de simulation de robotique mobile.

II Intelligence en essaim

II.1 Introduction à l'intelligence en essaim

En 1989, Beni et Wang (Beni & Wang, 1993) ont utilisé le terme intelligence d'essaim dans le domaine des systèmes robotiques cellulaires pour définir une collection de robots autonomes, non synchronisés et non intelligents coopérant pour atteindre un objectif global, tandis que dans le domaine de l'intelligence artificielle, elle était définie comme le comportement collectif de systèmes décentralisés et auto-organisés (naturels ou artificiels). Autrement dit, la robotique en essaim est définie comme un ensemble de nombreux individus simples qui interagissent avec d'autres individus et l'environnement en utilisant une gestion décentralisée et auto-organisée pour atteindre leurs objectifs **(Cruz et al.,2019)**. Bonabeau et al.

Ont étendu leur définition pour inclure toute tentative de concevoir des algorithmes ou des dispositifs distribués de résolution de problèmes inspirés par le comportement collectif des colonies d'insectes sociales et d'autres sociétés animales **(Bonabeau et al., 1999).**

1) Naissance et définitions

L'intelligence en essaim est une branche des techniques modernes d'intelligence artificielle étudiées et inspirées par les comportements collectifs des insectes sociaux, des animaux et des sociétés humaines (Li & Clerc, 2019), Et traite de la conception de systèmes multi-agents tels que l'optimisation et la robotique **(Charrier,2009)**.Puisque l'intelligence est une entité collective et non une entité isolée, l'intelligence en essaim peut donc être considérée comme un concept plus large d'intelligence car elle met l'accent sur le fait que l'intelligence doit être modélisée dans un contexte social, en raison de l'interaction les uns avec les autres (Li & Clerc, 2019).

La Figure 1.1 présente quelques exemples de systèmes qualifiés d'intelligents en essaim.

Banc de poisson **[1]**

Essaim de chauvesouris **[2]**

Essaim d'abeilles **[3]**

Essaim de fourmis **[4]**

Figure 1.1 : Exemples de systèmes qualifiés d'intelligents en essaim.

Définition 1.1 : '*L'intelligence en essaim est un domaine hautement bio-inspiré dont l'objectif est de modéliser au moyen de systèmes multi-agents, les mécanismes d'auto-organisation et d'adaptation observés dans les organismes vivants. Ces modèles donnent finalement naissance à des algorithmes qui simulent des phénomènes naturels dans des logiciels (sur un ordinateur) ou du matériel (avec des robots), ou servent de méta-heuristiques aux problèmes de l'intelligence artificielle'* **(Charrier,2009)**.

Définition 1.2 : '*D'un point de vue méthodologique, l'intelligence en essaim est un ensemble de solutions heuristiques inspirées des comportements des essaims d'animaux et capables d'offrir des solutions empiriques à de nombreux problèmes de calcul difficiles appartenant à plusieurs disciplines'* **(Zoghby et al., 2014)**.

Définition 1.3 : '*L'intelligence en essaim fait référence à une sorte de capacité de résolution de problèmes qui émerge dans les interactions de simples unités de traitement de l'information. Les unités de traitement de l'information qui composent un essaim peuvent être animées, mécaniques, informatiques ou mathématiques ; ils peuvent être des insectes, des oiseaux ou des êtres humains ; ils peuvent être réels ou imaginaires. Leur couplage peut avoir un large éventail de caractéristiques, mais il doit y avoir interaction entre les unités'* (Li & Clerc, 2019).

2) Propriétés des systèmes d'intelligence en essaim

L'intelligence en essaim est caractérisée par les propriétés suivantes (Mayeur, 2014).

- **Mise à l'échelle :** elle résout des problèmes de différentes tailles et le nombre d'individus impliqués n'affectera pas l'expérience.
- **Tolérance aux erreurs :** Par la redondance du système, et l'absence de contrôle centralisé, le système est aussi robuste que possible. Si un robot devient défectueux, il peut être remplacé par un autre robot. L'utilisation de capteurs qui ne peuvent capturer que des données locales peut également conduire à de mauvaises décisions lors de l'exécution des différents algorithmes. Cependant, la redondance du système conduit à compenser les erreurs, et même si un robot devait prendre une mauvaise décision, la majorité des robots seraient même capables de réagir correctement pour résoudre le problème.
- **Flexibilité :** C'est la possibilité pour un essaim de robots de résoudre différentes tâches dans différents environnements. Ceci s'inscrit dans les conséquences de la redondance des systèmes. Si

les mêmes robots sont utilisables pour différentes tâches et environnements, ils peuvent alors s'adapter à ceux-ci**.**

II.2 Interactions et transferts d'information dans les systèmes d'intelligence en essaim

Dans la nature, les interactions représentent le canal de communication entre les insectes et entre eux et l'environnement, ce qui signifie que chaque société a sa propre manière d'interaction. Les interactions permettent aux insectes de partager et d'obtenir des informations sur les conditions environnementales et des colonies. Il existe deux types de communication **(Cruz et al., 2019),** (Garnier, 2008) :

- **Transfert indirect d'information :** Les interactions indirectes sont la communication entre les insectes médiés par l'environnement. Certains individus modifient l'environnement et d'autres perçoivent cette modification, ajustant leurs comportements en conséquence. Cette réaction est un exemple de stigmatisation. Ce terme signifie la forme de communication indirecte dans laquelle chacun travaille sur l'environnement et d'autres individus qui découvrent certains changements dans l'environnement interagissent avec la stimulation. Ce processus conduit à une coordination quasi complète du travail d'équipe et peut nous donner l'impression que la colonie suit un plan précis **(Cruz et al., 2019),** (Garnier, 2008)**.**
- **Transfert direct d'information :** Consiste en une communication locale où aucune modification de l'environnement n'est requise. Les informations échangées par des interactions directes peuvent être de différents types, telles que contact physique, échange de fluides, signes visuels et acoustiques ou remarquer le comportement de leurs voisins ...etc. Pour que l'information circule sous plusieurs formes: visuelle, tactile ou auditive, mais cela ne dure pas dans l'environnement.

  De nombreux comportements d'auto-organisation dépendent des interactions directes entre les membres du groupe. Par exemple, des volées d'oiseaux se déplacent de manière cohérente et changent de direction soudainement mais simultanément, c'est le résultat d'une forte simulation comportementale couplée à un transfert direct d'informations **(Cruz et al., 2019),** (Garnier, 2008)**.**

II.3 Fonctions assurées par les systèmes d'intelligence en essaim

Dans un contexte biologique, les processus d'auto-organisation remplissent souvent des fonctions qui permettent au groupe ou à la société animale d'appréhender les divers éléments de son environnement. Il est possible de regrouper ces fonctions en trois catégories : coordination, collaboration et délibération.

1) **Coordination :** C'est l'organisation des tâches nécessaires dans l'espace et le temps pour résoudre un problème (Garnier, 2008). Cette fonction affecte la séquence temporelle et / ou la distribution spatiale des activités des individus pour atteindre un objectif donné. Par exemple, la coordination du travail pendant la construction du nid chez certaines espèces de guêpes sociales ou termites. En général, la coordination de l'activité des individus est la principale conséquence des processus d'auto-organisation qui sont au cœur des systèmes d'intelligence en essaim **(Garnier et al., 2007).**

2) **Collaboration :** La collaboration se produit lorsque des individus accomplissent ensemble une tâche qui ne peut être accomplie par un seul. Les individus doivent conjuguer leurs efforts pour résoudre avec succès un problème qui dépasse leurs capacités individuelles **(Garnier et al., 2007)**, c'est-à-dire la répartition des activités entre individus ou groupes d'individus spécialisés dans la réalisation d'une de ces activités, qui peut être établie sur la base des différences morphologiques interindividuelles : des individus ayant des capacités physiques différentes effectueront différentes tâches. Il apparaît que ce mécanisme basé sur l'expérience des individus est à l'origine de l'organisation du travail chez plusieurs espèces d'insectes sociaux (Garnier, 2008)**.**

3) **Délibération :** La délibération fait référence aux mécanismes qui se produisent lorsqu'une colonie fait face à plusieurs opportunités, qui fera un choix collectif ou une décision collective. Par exemple, la fourmi des espèces de « Lasius Niger » est en mesure de choisir la plus riche parmi plusieurs sources de nourriture, et parmi plusieurs chemins menant au chemin le plus court vers la source (Garnier, 2008)**.**

II.4 Applications de l'intelligence en essaim

Il existe des nombreuses applications de l'intelligence en essaim comme : Optimisation, robotique en essaim, data mining, les applications militaires, foule simulant, essaimage humain, art essaim, sécurité routière... *(MAHENDRA & PANDEY, 2016)***,** *(LI & CLERC, 2019)***,** Ci-dessous, nous présentons deux de ces exemples :

- **Résolution de problèmes d'optimisation :** Parmi la gamme des problèmes d'optimisation, on distingue les problèmes d'optimisation discrets, les problèmes de variables continues comme les problèmes d'identification qui consistent à minimiser une fonction d'erreur entre le modèle d'un système et des valeurs expérimentales. Pour les problèmes discrets, de nombreuses heuristiques ont été développées pour réduire les temps de calcul et aborder au mieux l'optimalité. Pour les problèmes de variables continues, des méthodes « d'optimisation globale » existent, mais elles deviennent

inefficaces dès que certaines propriétés de la fonction objectif ne sont plus vérifiées (par exemple si la fonction objective n'est pas convexe) (Charrier, 2009).

- **Robotique d'essaim :** Il s'agit d'un nouveau domaine de recherche concerné par la conception, le contrôle et la coordination de systèmes multi-robots. La robotique en essaim est définie comme l'étude de la façon dont un grand nombre d'agents incarnés physiquement relativement simples peuvent être conçus de telle sorte qu'un comportement collectif souhaité émerge des interactions locales entre les agents et entre les agents et l'environnement (Li & Clerc, 2019). La robotique est l'une des applications ciblées par l'intelligence en essaim ; robotique d'essaim offre un champ d'expérimentation et de validation stimulant pour les algorithmes d'intelligence en essaim (Charrier, 2009).

II.5 Principaux algorithmes issus de l'intelligence des essaims

Plusieurs approches inspirées des comportements sociaux ont été proposées pour résoudre différents problèmes. Citons et définissions par la suite les plus popularités des algorithmes : l'optimisation des colonies de fourmis (ACO), l'optimisation des essaims de particules (PSO), la colonie d'abeilles artificielles (ABC) et l'algorithme lucioles (FA), l'algorithme de foraging bactérienne (BFA) et l'algorithme de libellule (DA) ci-dessous :

1) **L'optimisation des Colonies de Fourmis (Ant Colony Optimization- ACO) :** Marco Dorigo et ses collègues ont introduit l'optimisation des colonies de fourmis (ACO) au début des années 1990, comme une solution méta-heuristique inspirée de la nature pour problèmes d'optimisation combinatoire difficiles **(Patnaik et al., 2017)**. La source d'inspiration de l'ACO est le comportement des fourmis réelles, qui utilisent les phéromones comme moyen de communication (Dorigo & Stützle, 2019). Il est basé sur le comportement foraging de fourmis, et l'objectif est de trouver le chemin optimal entre la source de nourriture et leur nid. Pour utiliser l'ACO, le problème ciblé doit être transformé en un problème de graphe pondéré et l'objectif sera de rechercher le meilleur chemin sur le graphe. Les agents des fourmis utilisent un processus stochastique pour

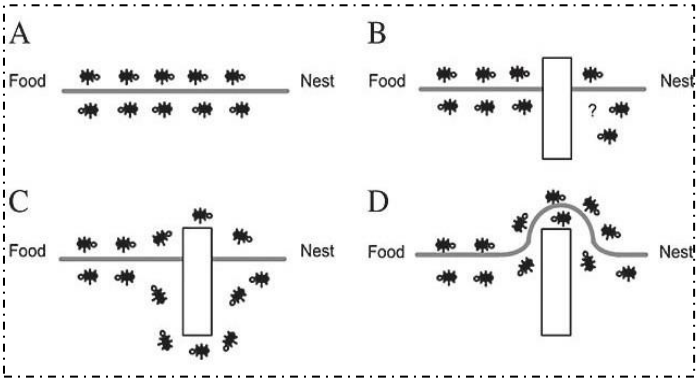


Figure 1.2 : Un exemple d'exécuter de ACO **(NADIPALLY, 2019)**

construire la solution en traversant le graphique, et ce processus est contrôlé par un modèle de phéromone composé d'un ensemble de paramètres associés à chaque composant du graphique.

2) **Optimisation par Essaims Particulaires (Particle swarm optimization- PSO) :** L'optimisation des essaims de particules (PSO) est une technique de calcul évolutif développée par Kennedy et Eberhart en 1995 (Li & Clerc, 2019). C'est une approche évolutive qui s'inspire du comportement social projeté par les individus dans un troupeau d'oiseaux ou un banc de poissons. Ce comportement social des particules individuelles est influencé par sa propre expérience passée ainsi que par l'expérience passée de son voisinage **(Patnaik et al., 2017).** L'optimiseur d'essaim de particules se compose d'un certain nombre de particules, qui orbite autour et recherchent l'espace pour le meilleur emplacement. Les individus communiquent directement ou indirectement les uns avec les autres les directions de recherche. PSO est une technique de recherche simple mais puissante, avec peu de paramètres à régler et facile à mettre en œuvre **(Jevtić et al.,2007).**

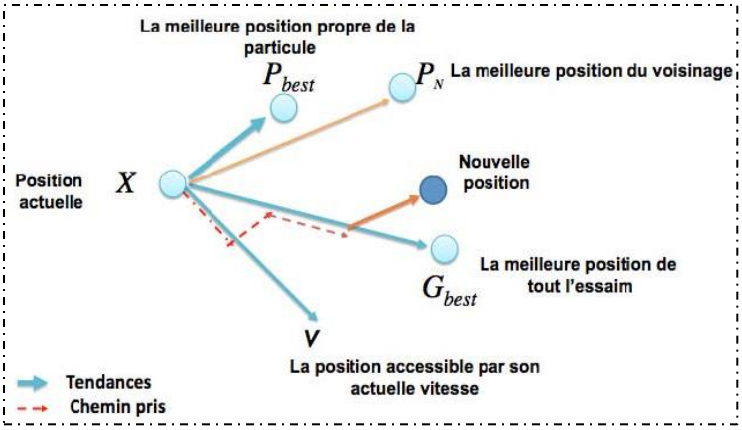


Figure 1.3 : Principe de fonctionnement du PSO **(ZEMZAMI, ET AL., 2017)**

3) **Colonies d'Abeilles Artificielles (Artificial Bee Colony- ABC) :** L'algorithme est défini comme un méta-heuristique qui adopte la technique employée par un essaim intelligent d'abeilles pour identifier leur source de nourriture. La nature des abeilles est étudiée en fonction de leur communication, de la sélection de l'emplacement du nid, de l'accouplement, de l'attribution des tâches, de la reproduction, de la danse, du placement des phéromones et du mouvement pour modifier ensuite l'algorithme en fonction des exigences du

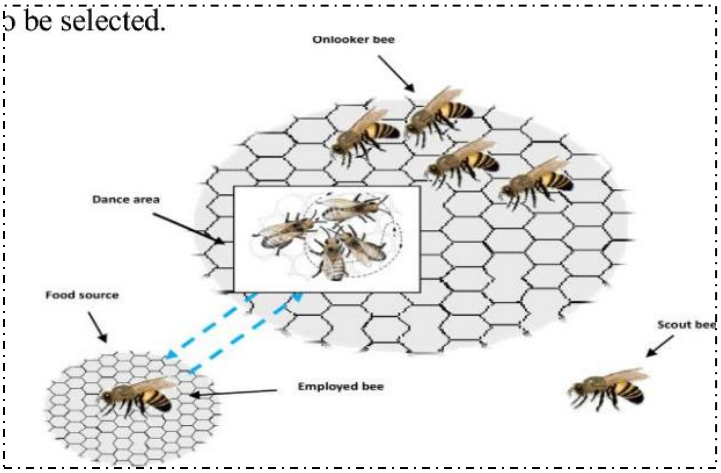


Figure 1.4 : Les types d'abeille dans un ABC et leur fonctionnement **(KAUR & SHARMA, 2017)**

problème. L'algorithme de la colonie d'abeilles artificielles effectue une optimisation en recherchant de manière itérative la solution là mieux adaptée parmi un grand nombre de données tout en essayant de résoudre des problèmes critiques **(Patnaik et al., 2017).**

4) **Algorithme des Lucioles (Firefly Algorithm- FA) :** L'algorithme Firefly (FA) a été développé pour la première fois par Xin-She Yang fin 2007 et 2008 à l'université de Cambridge sur la base du comportement de clignotement des lucioles (Yang, 2014)**.** FA suppose qu'une solution d'un problème d'optimisation est codée comme l'emplacement de l'agent / luciole, tandis que la fonction objectif est codée comme l'intensité lumineuse. Dans FA, il y a deux problèmes importants qui font le succès de l'algorithme : (1) les variations dans l'intensité lumineuse, et (2) la bonne formulation de l'attractivité, qui est déterminée par la fonction de luminosité, qui à son tour est associée à la fonction objective **(Srivatsava et al., 2013).**

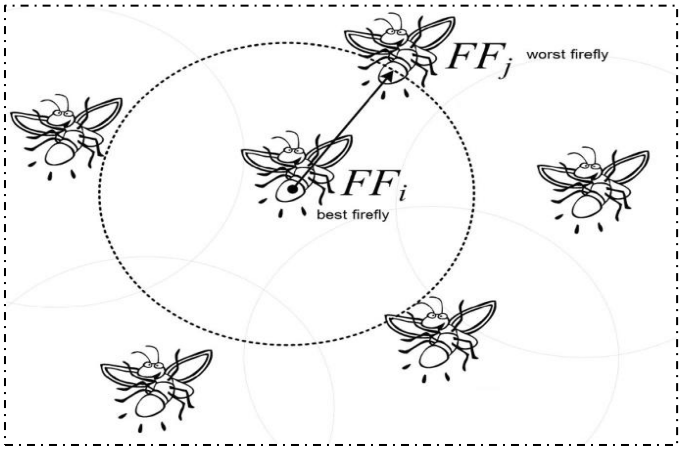


Figure 1.5 : Explication for FA **(Ali et al.,2019)**

5) **Algorithme de Foraging Bactérienne (Bacterial Foraging Algorithm- BFA) :** L'algorithme de recherche de nourriture bactérienne est inspiré par le comportement de recherche de nourriture de groupe de bactéries telles que E. Coli et M. Xanthos. Le BFA est inspiré par le comportement chimiotactique des bactéries qui percevront les gradients chimiques dans l'environnement (tels que les nutriments) et se déplaceront vers ou loin de signaux spécifiques (Mahendra & Pandey, 2016).

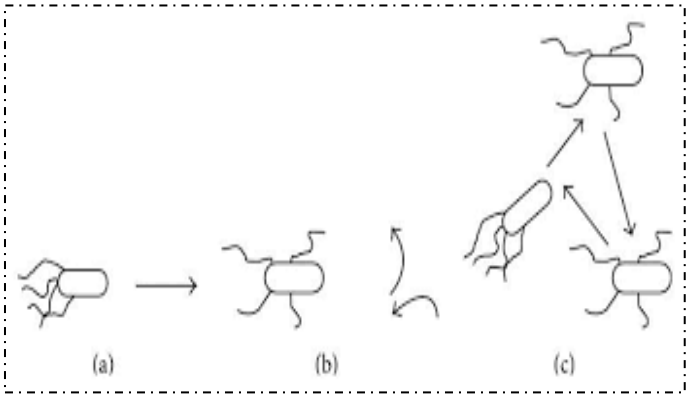


Figure 1.6 : Quelque modèles de comportement de BFA **(LARFI & AISSET, 2016)**

6) **Algorithme de libellule (Dragonfly Algorithm « DA ») :** Introduire par Seyedali Mirajalli en l'idée principale de l'algorithme libellule est dérivée des comportements statiques et dynamiques des libellules dans la nature. On peut dire que ces deux comportements sont très similaires aux deux phases importantes de l'optimisation, à savoir la recherche de proies (exploration) et l'attaque des proies

(exploitation). Les étapes importantes du comportement d'essaimage des libellules sont décrites ci-dessous (Abedi & Gharehchopogh, 2020).

- *Séparation (S)* dans cette étape, les essaims sont séparés des autres individus, ce qui évite les conflits avec les voisins.
- *Alignement (A),* la vitesse de chaque individu est adaptée aux autres.
- *La cohésion (C)* concerne l'attraction de l'essaim vers le centre de l'équipe d'essaims.

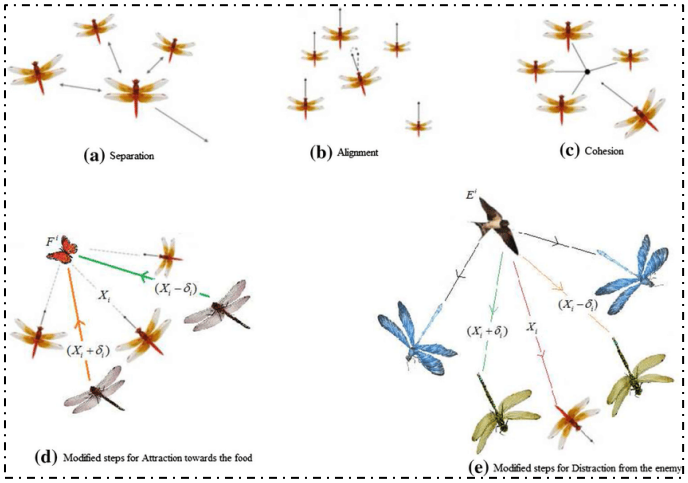


Figure 1.7 : Principe de fonctionnement de DA **(Suresh et al., 2019)**

- *L'attraction vers l'origine* de la nourriture est mathématiquement montrée par (F).
- *Distraction de l'ennemi (E).*

III Robotique en essaim

Il existe de nombreuses tâches dangereuses et mortelles que les humains ne peuvent pas effectuer, telles que les applications industrielles, les interventions en cas de catastrophe et les applications militaires. Pour cette raison, les robots dans des environnements difficiles sont une application essentielle qui peut fournir des services sûrs et optimaux au lieu de laisser les humains opérer dans un environnement dangereux. Soit en créant un robot unique extrêmement puissant et coûteux, soit en déployant un groupe de robots simples qui, ensemble, accompliront des tâches plus efficaces qu'un seul robot. Cette dernière approche est connue sous le nom de robotique en essaim.

Les robots en essaim sont une méthode relativement nouvelle pour les systèmes Multi-Robots, inspirés par le comportement des animaux sociaux tels que les fourmis, les oiseaux, les poissons, etc., un groupe peut effectuer une tâche complexe. En simulant le comportement collectif d'animaux sociaux, les robots en essaim peuvent mettre en œuvre des scénarios complexes à l'aide de robots simples **(Huang et al., 2019).**

La Figure 1.8 montre quelques exemples des essaims de robots.

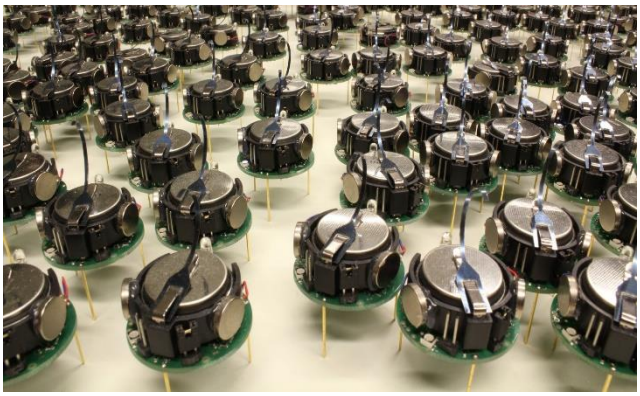
	
Swarm micro-robots **[6]**	Swarm drones **[7]**
	
Swarm nano-bots **[8]**	Swarm cyborg **[9]**

Figure 1.8 : Exemples de systèmes qualifiés d'essaim de robots

III.1 Définitions

Définition 1.4 : *'La robotique en essaim est l'étude de la façon de créer des groupes de robots qui fonctionnent sans dépendre d'une infrastructure externe ou de toute forme de contrôle centralisé. Dans un essaim de robots, le comportement collectif des robots résulte de l'interaction locale entre les robots et l'environnement dans lequel ils interagissent'* (Dorigo & Gambardella, 1997).

Définition 1.5 : *'La robotique en essaim est définie avec les principes qu'un système d'essaim doit avoir un grand nombre de robots, les tâches doivent être résolues et améliorées en utilisant un système d'essaim et que les robots échangent des informations locales par des distances de communication limitées'* **(Huang et al., 2019)**.

Définition 1.6 : *'La robotique en essaim est l'étude de la façon dont un grand nombre d'agents incarnés physiquement relativement simples peuvent être conçus de telle sorte qu'un comportement collectif souhaité émerge des interactions locales entre les agents et entre les agents et l'environnement'* (Sahin, 2004).

III.2 Caractéristiques

La robotique en essaim doit être caractérisée avec ces principales caractéristiques observées dans la nature (Li & Clerc, 2019)**, (Zoghby et al., 2014),** (Yogeswaran & Ponnambalam, 2010) **:**

- **Parallélisme :** les robots peuvent accomplir une tâche donnée plus rapidement qu'un seul robot en divisant la tâche en sous-tâches et en les exécutant simultanément.
- **Robustesse :** le système est toujours en mesure de fonctionner malgré les perturbations de l'environnement ou le dysfonctionnement de certains individus.
- **Évolutivité :** l'essaim est capable d'opérer sous différentes tailles de groupe et avec un grand nombre d'individus.
- **Hétérogénéité :** Puisqu'un groupe de robots peut être hétérogène, il peut utiliser des robots spécialistes dont les propriétés physiques leur permettent d'effectuer efficacement certaines tâches bien définies.
- **La flexibilité :** les individus de l'essaim peuvent coordonner leurs comportements pour accomplir diverses tâches.
- **Tâches complexes :** Les tâches peuvent être intrinsèquement trop complexes (ou impossibles) pour un seul robot à accomplir ou à exécuter, alors qu'un essaim est capable.
- **Alternative pas chère :** Construire et utiliser plusieurs robots simples peut être plus facile, moins cher, plus flexible et plus tolérant aux pannes que d'avoir un seul robot puissant pour chaque tâche distincte.

III.3 Applications

1) **Agrégation :** L'agrégation est utilisée pour placer les robots dans un essaim suffisamment rapproché et peut être utilisée comme point de départ pour effectuer des tâches supplémentaires complexes, telles que le transport collectif où les objets

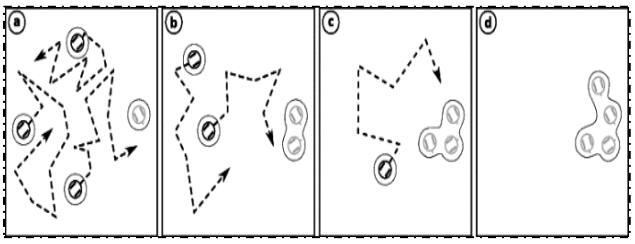


Figure 1.9 : un exemple de agrégation **(CAMPO & DORIGO, 2020)**.

d'intérêt doivent être transportés par plusieurs robot **(Liekna et al., 2014).**

Figure 1.10 : une simulation de exploration **(FALCONI, 2009)**.

2) **Exploration :** L'objectif d'un processus d'exploration est de couvrir l'ensemble de l'environnement dans un délai raisonnable ; il s'agit de découvrir des zones auxquelles les humains ne peuvent pas accéder facilement.

3) **Déplacements coordonnés :** Les robots adoptent ici une direction de mouvement commune au sein d'un groupe où ils doivent estimer à plusieurs reprises la direction de mouvement des autres membres du groupe afin qu'elle reste constante. La tâche des mouvements coordonnés représente le mouvement tout en préservant l'entraînement et est aussi appelée flocage (flocking). Ceci est utile dans les applications impliquant des groupes de mouvements de robots car la préservation de la formation permet d'éviter les collisions entre robots et sert de mécanisme de navigation **(Liekna et al., 2014).**

Figure 1.11 : une simulation de flockage **(DUCATELLE, ET AL., 2014)**

4) **Transport coopératif :** La tâche de transport collectif implique la coopération d'un robot pour transporter collectivement un objet, étant donné que le transport d'un seul objet nécessite plus d'un robot **(Liekna et al., 2014).**

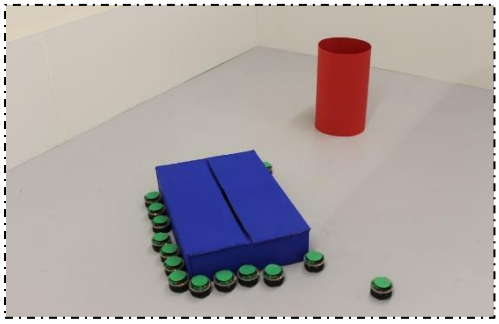


Figure 1.12 : une simulation de transport collectif **(CHEN, ET AL.,**

5) **Foraging :** La recherche de nourriture (foraging) est connue comme tâche de récupération ou de collecte de proies **(Liekna et al., 2014).** Les applications potentielles de la recherche de nourriture dans un scénario réel sont les opérations de recherche et de sauvetage, les opérations de nettoyage des déchets toxiques **(Jeuvtić et al.,2007).**

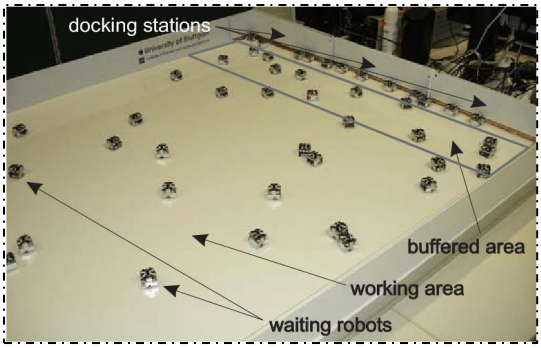


Figure 1.13 : une simulation de foraging **(KORNIENKO & KORNIENKO, 2011)**

6) **Recherche et sauvetage :** Des essaims de robots envoyés dans des endroits où l'aide n'a pas pu être obtenue, alors ils recherchent et sauvent des vies. La recherche et sauvetage (SAR) est l'un des principaux problèmes de gestion des catastrophes ; car il s'agit d'une tâche difficile, limitée dans le temps et qui nécessite à la fois l'exploration de l'arène et la détermination de l'emplacement exact des victimes sans compromettre la sécurité des sauveteurs **(Din et al., 2018).**

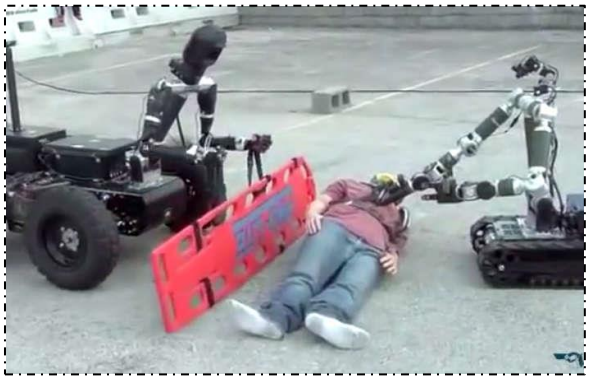


Figure 1.14 : une simulation de sauvetage.

Il existe bien d'autres applications de la robotique en essaim, citons par exemple **(Bahel et al., 2020),** (Mahendra & Pandey, 2016) **:**

- Sciences spatiales : La NASA utilisait auparavant la robotique d'essaim pour alimenter les missions d'exploration de l'espace (Mahendra & Pandey, 2016).
- Médicament : Les experts prévoient d'avoir des micro-robots avancés qui pénètrent profondément dans le corps et tuent toutes les cellules nocives à leur entrée.
- Pollinisation autonome d'un champ de cultures (Mahendra & Pandey, 2016).
- Surveillance et défense militaire : Les experts militaires estiment que les véhicules aéronautiques bioniques inspirés de la technologie de renseignement sur les essaims deviendront applicables dans quelques années. Il peut être prévu que des abeilles ou des cafards avec du matériel de reconnaissance et des bombes apparaîtront dans une guerre future (Tan, 2013)**.**
- Cartographie météorologique et climatique haute résolution.
- Surveillance du trafic.
- Nettoyage et détection des déversements d'hydrocarbures : Le « Senseable City Lab » du MIT a développé une flotte de robots à faible coût absorbant le pétrole appelés Seaswarm pour l'écrémage des océans et l'élimination du pétrole. Un robot nanomatériau peut absorber du pétrole jusqu'à 20 fois son poids. Le système fournit une solution autonome pour la protection de l'environnement océanique (Tan, 2013).

III.4 Avantage de la robotique en essaim

Quelques avantages de la robotique en essaim (Charrier, 2009), **(Huang et al., 2019)**:

- Le système peut être évolutif en augmentant le nombre de robots,
- La robustesse est accrue du fait de la division des tâches,
- Les fonctions du système peuvent être facilement modifiées en ajoutant ou en supprimant différents types de robots.
- Un système d'essaim a de grandes capacités d'adaptation, quel que soit son environnement,
- Le système d'essaim peut résoudre des problèmes complexes en raison de ses multiples unités de calcul, bien au-delà des problèmes que les entités individuelles peuvent résoudre.

III.5 Plateformes de simulation de robotique en essaim

1) Autonomous Robots Go Swarming ARGoS:

ARGoS est un simulateur multi-robots capable de simuler de grands essaims hétérogènes de robots, ARGoS est le premier simulateur multi-robots à la fois efficace (performances rapides avec de nombreux robots) et flexible (hautement personnalisable pour des expériences spécifiques) (Pinciroli, et al., 2012)**.**

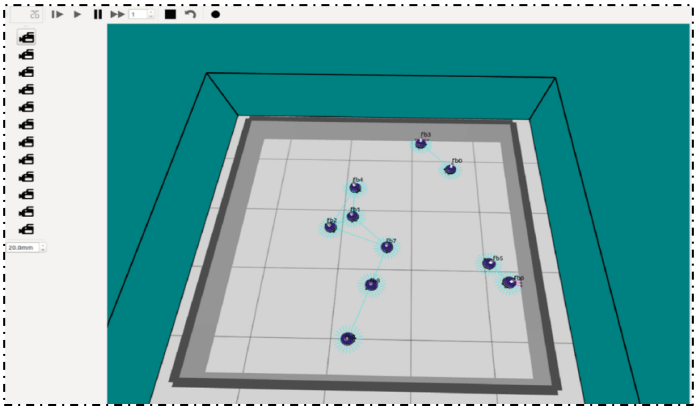


Figure 1.15 : l'interface du ARGoS **(O'keeffee et al., 2017)**

2) Player / Stage : Stage est souvent utilisé comme module de plugin Player, fournissant des populations de périphériques virtuels pour Player. Les utilisateurs écrivent les contrôleurs de robot et les algorithmes de capteur en tant que « clients » sur le « serveur » du lecteur. Stage permet un prototypage rapide de contrôleurs destinés à de vrais robots. Stage permet également des expériences avec des appareils robotiques réalistes que vous n'avez pas **[11].**

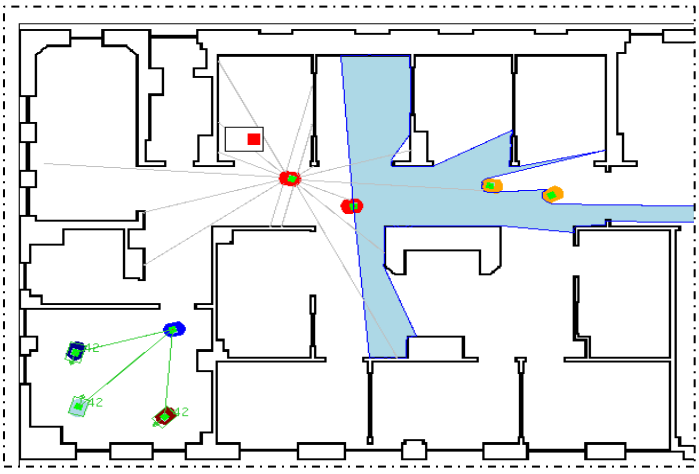


Figure 1.16 : le simulateur Player/Stage **[10]**

3) **Webots :** Webots est un progiciel de simulation de robot mobile professionnel. L'utilisateur peut ajouter des objets passifs simples ou des objets actifs appelés robots mobiles. Ces robots peuvent avoir différents schémas de locomotion (robots à roues, robots à pattes ou robots volants). L'utilisateur peut programmer chaque robot individuellement pour présenter le comportement souhaité. Webots contient un grand nombre de modèles de robots et d'exemples de programme de contrôleur pour aider les Utilisateurs à démarrer **[13]**.

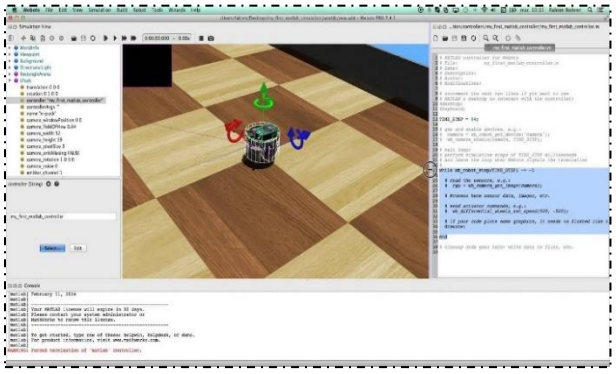


Figure 1.17 : le simulateur Webots **[12]**

4) **Gazebo :** Gazebo est un simulateur dynamique 3D avec la capacité de simuler avec précision et efficacité des populations de robots dans des environnements intérieurs et extérieurs complexes. Bien que similaire aux moteurs de jeu, Gazebo offre une simulation physique à un degré de fidélité beaucoup plus élevé, une suite de capteurs et des interfaces pour les utilisateurs et les programmes **[15].** Gazebo est un logiciel gratuit, publié sous la licence publique GNU. Vous êtes libre d'utiliser, d'étendre et de modifier selon vos besoins **[16].**

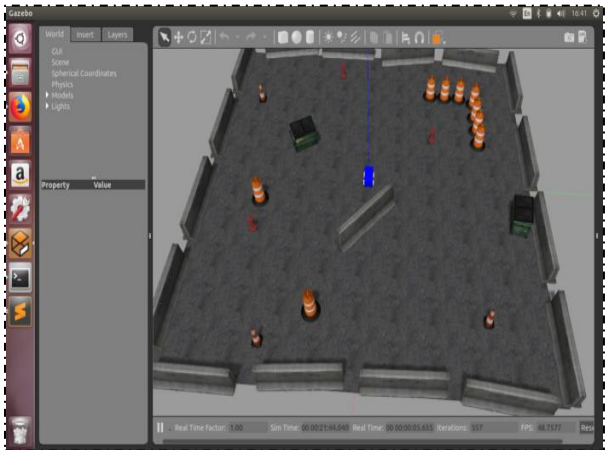


Figure 1.18 : le simulateur Gazebo **[14]**

IV Conclusion

Ce chapitre a été divisé en deux parties. Dans la première, nous avons présenté quelques définitions du domaine de l'intelligence en essaim. Dans la deuxième partie, nous avons défini le domaine de la robotique en essaim ainsi que certains comportements de base qui en découlent. Nous avons terminé cette partie avec quelques-unes des plateformes de simulation pour la robotique en essaim.

Chapitre 2 : Recherche et Sauvetage Multi-Robots – Synthèse des travaux

I Introduction

Il y a beaucoup de catastrophes dans notre vie telles que : tremblements de terre, ouragans, tsunami et attaques terroristes. Ces catastrophes entrainent les blessures, les handicaps...etc. L'évaluation correcte de la situation joue un rôle clé lors de ces opérations. Cependant, réussir dans un problème aussi difficile et dangereux nécessite des informations fiables, une bonne organisation et une utilisation efficace des ressources. Les robots peuvent aider en remplaçant les humains dans de telles situations, ce qui minimise les risques pour le personnel de recherche et de sauvetage, tout en augmentant les taux de survie des victimes, en déployant des équipes de robots collaboratifs, en réduisant les besoins personnels et en accédant à des zones autrement inaccessibles sont les avantages des robots de sauvetage.

Dans ce chapitre, on se concentre sur l'application de recherche et le sauvetage, nous commençons par une petite définition de cette application. Puis nous synthétisons les travaux reliés et nous terminerons par une comparaison des travaux à travers des critères que nous avons proposé et jugé importants.

II Recherche et sauvetage Multi-Robots

II.1 Définition

La recherche et le sauvetage (SAR) est l'un des principaux problèmes de la gestion des catastrophes ; car c'est une tâche difficile, une tâche limitée dans le temps et qui nécessite à la fois l'exploration de l'arène et la détermination de l'emplacement exact des victimes sans compromettre la sécurité des sauveteurs.

Dans la robotique mobile, l'exploration et la localisation sont toujours des défis, mis à part l'exigence de le faire avec une contrainte de temps, même si cela présente de nombreux avantages par rapport aux robots complexes basés sur des modèles dans des applications telles que la recherche et le sauvetage, l'exploration et la couverture spatiales, la surveillance, le transport et le déminage.

III Recherche et sauvetage Multi-Robots – Synthèse des travaux

Dans cette section, nous synthétisons un ensemble d'articles reliés, et nous les comparons qualitativement dans un tableau de comparaison.

1) Cooperative Search and Rescue with Swarm of Robots Using Binary Dragonfly Algorithm (Abuomar & Al-Aubidy, 2018)

Abuomar et Al-Aubidy proposent dans (Abuomar & Al-Aubidy, 2018), un algorithme de recherche et de sauvetage avec un essaim de robots. L'algorithme fait une amélioration de l'algorithme d'optimisation de libellule binaire (Optimization binary dragonfly) en concentrant sur deux comportements : l'évitement d'obstacles et de contraintes de communication.

Dans cet article, l'algorithme binaire de libellule (BDA) a été discuté et développé ; le BDA est une technique d'optimisation méta-heuristique classée comme un type d'algorithmes évolutifs. Il fonctionne comme un comportement statique et dynamique et provient des libellules dans la nature. Ils sont également tenus de changer leurs poids de manière adaptative pour passer de l'exploration à l'exploitation de l'espace de recherche.

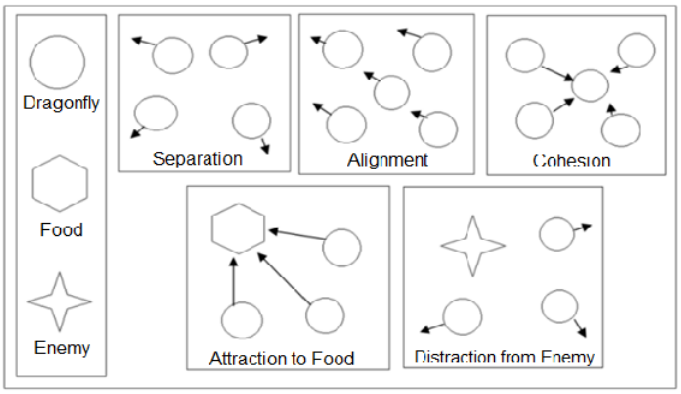


Figure 2.1 : Paramètres principaux de l'algorithme libellule (Abuomar & Al-Aubidy, 2018)

Les cinq comportements de base de BDA sont : *séparation* qui concerne l'évitement d'obstacles et tout autre individu ; *Alignement* qui concerne la correspondance de vitesse des individus ; *Cohésion* qui concerne le rapprochement vers un centre d'une zone ; *Attraction* qui concerne l'attraction a une zone d'intérêt (moniteur) ; et *Distraction* qui concerne le comportement de fuir de toute contrainte qui influence négativement le suivi des individus.

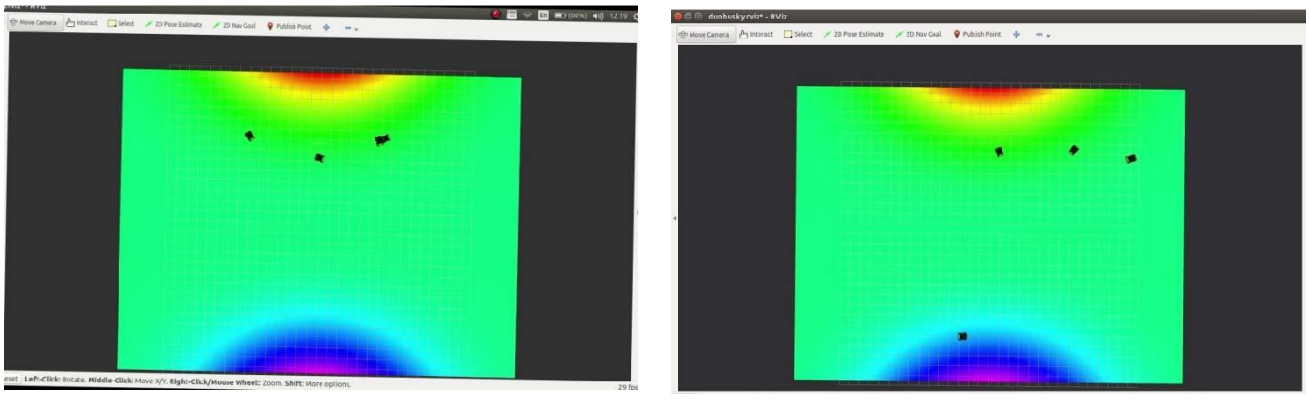
L'algorithme mis en œuvre a été testé sous le Toolbox virtual « SIMROBOT » sous Matlab pour valider la solution proposée où un robot avec algorithme victime peut être ajouté à l'environnement de travail, et peut être considéré comme la solution optimale. L'environnement ($300\ x\ 300\ m$) mis en œuvre a des obstacles statiques et dynamiques et distribuées aléatoirement. Le nombre initial de robots est de 5 robots et le nombre maximal de robots est de 15 robots. L'algorithme RBDA a été testé sur des fonctions de Benchmark (Sphere, Rosenbrock, De Jong's, Griewank, Rastrigin, Ackley) et comparé avec l'algorithme

BDA. Les résultats montrent que RBDA minimise l'erreur pour converger à la solution optimale. Les résultats simulés ont montré que l'algorithme RBDA a une meilleure précision de convergence pour atteindre la solution optimale en un temps plus court, et que l'augmentation du nombre de robots diminuera le nombre de tours, ce qui réduira le temps nécessaire pour atteindre la solution optimale.

2) Search and Rescue Operations Using Robotic Darwinian Particle Swarm Optimization (Kumar et al., 2017)

Kumar et al. **(Kumar et al., 2017)** Ont implémenté le Robotic Darwinian Particle Swarm Optimization (RDPSO) c'est un algorithme d'exploration qui surmonte la limitation de convergence sur plusieurs cibles, c'est considéré comme une extension de PSO. Il permet la création dynamique de groupes au sein de l'essaim, ou d'autres essaims, en utilisant un mécanisme de récompense et de punition. Par conséquent, RDPSO permet une diminution de la quantité d'informations requises à partager entre les robots.

Tous les robots qui utilisent l'algorithme RDPSO exploreront initialement (solitairement ou en petits groupes) dans des directions aléatoires jusqu'à ce qu'ils reçoivent un signal, par exemple le son de la voix humaine ou l'intensité des incendies. Une fois un signal reçu est connu, le robot compare l'intensité du signal avec tous ses partenaires pour voir quel robot a détecté l'intensité maximale du signal. Par exemple, le robot le plus proche de la victime recevra la valeur maximale d'intensité. Ce robot est traité comme le robot le plus performant de l'essaim et sera récompensé en lui donnant un robot partenaire. D'autres robots socialement exclus marchent dans des directions aléatoires jusqu'à ce que toute la zone soit couverte. De cette façon, des opérations de recherche et de sauvetage sont effectuées.



(a) : utilisation de PSO **(b) :** utilisation de RDPSO

Figure 2.2 : environnement de simulation dans la phase finale **(Kumar et al., 2017)**

Les auteurs ont simulé des missions de recherche et de sauvetage dans la vie réelle avec plusieurs cibles statiques et des robots d'essaim. Les victimes étaient supposées être statiques dans tous les essais. Ces robots n'ont aucune connaissance préalable de la localisation des victimes ou des incendies dans l'environnement et n'ont que la connaissance de leurs positions initiales, vitesses initiales et environnement à explorer. Deux algorithmes d'exploration ont été utilisés pour effectuer des opérations SaR et ont évalué leurs performances dans des opérations SaR avec plusieurs cibles. La simulation est présentée à l'aide de Robot Operating System (ROS) et Gazebo, avec une visualisation dans Rviz utilisant des cartes d'intensité qui imitent des scénarios du monde réel. Le résultat expérimental a montré que RDPSO a de meilleures performances par rapport à RPSO pour les opérations SaR à cibles multiples.

3) Behavior-based swarm robotic search and rescue using fuzzy controller (Din et al., 2018)

Din et al. **(Din et al., 2018)** Ont concentré sur le problème de la recherche et du sauvetage collectifs dans un site post-catastrophe en utilisant un modèle basé sur le comportement d'essaim de robots relativement petits avec des capacités de communication et de calcul limitées. Ce mécanisme d'exploration et de couverture basé sur le comportement est étendu vers une architecture basée sur l'essaim et l'application de la logique floue, où un leader virtuel conduira l'essaim vers une zone inexplorée, il est sélectionné dynamiquement.

Les performances d'un essaim dépendent en grande partie des règles de contrôle locales des robots individuels. Chaque robot de l'essaim a un système de contrôle embarqué, un module de communication sans fil, un système de capteur (laser et ultrasons) et un contrôleur de mouvement. Les robots doivent être dynamiques, adaptatifs et flexibles. Un mécanisme autonome de recherche d'objectifs dynamiques, utilisant une approche basée sur le comportement, est conçu pour rechercher des cibles (victimes) sous la direction d'un leader virtuel dynamique bio-inspiré, et la logique floue a été utilisé pour concevoir des modules comportementaux constituants à savoir éviter le comportement passé, évitement d'obstacles, alignement et cohésion inter-robots. Dans ce qui suit, une description des sous-comportements d'essaim est présentée :

– *Contrôle de cohésion-séparation :* Les champs de potentiel d'attraction et de répulsion dépendent de la distance entre les robots. Le champ potentiel d'attraction est plus fort à de plus grandes distances, tandis que la répulsion domine lorsque les robots sont plus proches. Il existe une distance *d* à laquelle ces deux champs potentiels sont égaux.

- *Comportement d'alignement :* Le comportement d'alignement consiste à diriger un robot vers le pas moyen de ses voisins. Ce comportement peut être obtenu en contrôlant la vitesse de chaque robot presque égal à la vitesse moyenne de l'essaim.
- *Éviter les comportements passés :* ce comportement consiste à éviter les visites répétées du même emplacement et à trouver de nouveaux emplacements non visités.
- *Localiser le mécanisme d'espace libre (LFS) :* L'objectif de l'LFS est d'explorer l'environnement sans visiter la même zone à plusieurs reprises. L'un des robots de première ligne est sélectionné comme navigateur, dont le travail consiste à exécuter le comportement de localisation d'espace libre. Ce comportement localise le plus grand espace libre vers lequel un essaim se déplace tout en minimisant les visites répétées. Les emplacements précédemment visités sont détectés en évitant les comportements passés, et les cellules visitées sont traitées comme un obstacle.
- *Comportement d'exploration du leader virtuel :* Le leader virtuel se déplace en fonction du comportement de recherche d'objectifs, qui le guide et le conduit efficacement de l'objectif actuel vers le nouvel objectif. La recherche d'objectif basée sur le flou dans le comportement a deux entrées ; la distance entre le but actuel et le nouveau but, appelée distance du but (GD), et la différence entre l'angle actuel du leader virtuel et l'orientation souhaitée pour atteindre le nouveau but, appelée erreur d'orientation (OE).

L'ensemble flou linguistique pour le GD contient des valeurs très grandes, grandes, moyennes, petites et très petites ({VL, L, M, S, VS}). L'ensemble flou linguistique de l'OE est représenté par positif grand, positif, positif petit, proche de zéro, négatif petit, négatif, grand négatif ({PL, P, PS, NZ, NS, N, NL}), la fonction d'appartenance de sortie des variables linguistiques ont sept valeurs linguistiques de grande dispersion, dispersion, petite dispersion, pas de changement, petite attraction, attraction et grande attraction représentées par un ensemble logique {LD, D, SD, NC, SA, A, LA}. Ce comportement devrait définir le cap du leader virtuel vers le nouvel objectif.

- *Mécanisme d'évitement d'obstacles (OA) :* chaque robot de l'essaim possède un capteur de distance laser pour détecter la distance de l'obstacle.
- *Comportement de mise à l'échelle de l'essaim :* ce comportement dépend de la localisation de l'espace libre, évite le passé et évite les comportements d'obstacle.

La simulation est faite à l'aide du simulateur ARGoS. Un essaim de robots homogènes est déployé dans l'un des coins qui explore la région urbaine désastreuse pour rechercher les victimes. Cet environnement

qui contient divers obstacles et débris, est réparti en neuf grilles de tailles égales, et un nombre égal de victimes sont réparties au hasard dans chaque grille en utilisant une technique d'échantillonnage stratifié, et la détection des victimes dépend du niveau de gravité. On suppose que les victimes sont statiques, car elles les considèrent comme piégées. Les robots se déplacent en troupeau en utilisant un mécanisme de sélection de leader dynamique pour diriger l'essaim vers des régions inexplorées, en évitant les visites répétées dans l'environnement. Les paramètres utilisés : La taille de l'espace de recherche est 100 × 100 unités, nombre de robots : 12, temps d'échantillonnage 100 ms, portée d'interaction maximale 3 unités, temps de simulation 25 min, Rayon dans lequel la victime peut être détectée 3 unités. Pour faciliter l'évaluation, quinze répétitions de chaque expérience ont été effectuées pour chaque condition pendant 25 min chacune, comprenant 370 essais expérimentaux (même en cas d'échec du leader). Il a été observé que le modèle proposé était plus résilient et qu'il fonctionnait mieux par rapport à plusieurs robots.

4) A Real-Life Robotic Application of the Particle Swarm Optimization Algorithm (Greenhagen et al., 2016)

Greenhagen et al. **(Greenhagen et al., 2016)** Ont décrit l'effort d'utiliser l'algorithme PSO pour contrôler un petit essaim de robots dans une application de recherche et sauvetage. L'algorithme a été légèrement modifié par rapport à son format d'origine pour mieux s'adapter aux applications robotiques réelles. Un mini-essaim de trois robots est utilisé pour rechercher un point d'intérêt (cible) dont l'emplacement n'est pas connu. Afin d'augmenter la difficulté du problème, une autre cible moins adaptée à la solution est utilisée pour étudier la capacité des robots à faire la distinction entre les deux cibles.

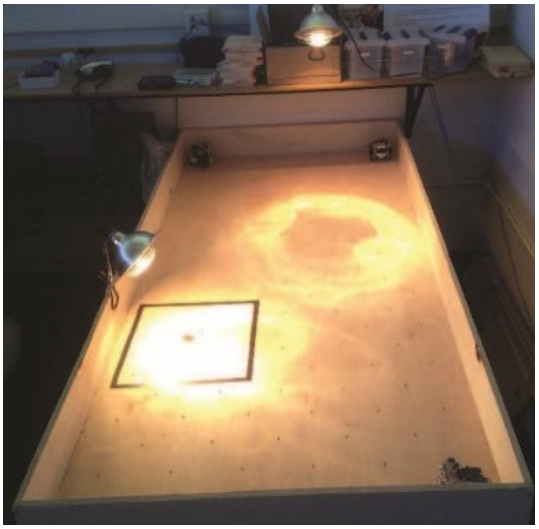


Figure 2.3 : L'arène réelle et la carte de chaleur correspondante **(Greenhagen et al., 2016)**

Dans cette étude, l'objectif était de remplacer les agents virtuels traditionnellement utilisés pour implémenter une simulation informatique basée sur PSO par des agents physiques qui naviguent réellement dans un espace de recherche réel à la recherche d'un objet (robots autonomes). Ils implémentent la méthode PSO sur un petit « essaim » de robots qui agira comme des « particules » parcourant un espace de recherche bidimensionnel à la recherche d'une solution optimisée.

Les auteurs suivent la configuration expérimentale suivante :

- *L'environnement de test :* une arène rectangulaire en bois de 8 x 4 pouces a été fabriquée pour servir d'environnement de test pour les robots. Dans cette arène, deux lampes chauffantes de 200 watts ont été utilisées pour créer des points chauds maximaux locaux et mondiaux. L'arène ainsi que les deux lampes sont représentées dans la Figure 2.22.
- *Les robots :* Les trois robots utilisés pour effectuer la recherche sont des « Propeller Activity Bot robots de Parallax, Inc ». Les robots incluent des platines montées permettant l'ajout de circuits auxiliaires tels que les circuits de détection de température ainsi que des modules de communication et des capteurs de proximité ultrasonore.
- *Communication inter-robots :* L'une des caractéristiques importantes de l'algorithme PSO est qu'il s'agit d'une méthode de contrôle décentralisée. En d'autres termes, les robots n'ont pas besoin de communiquer avec une station de base et ils ne se rapportent ni ne reçoivent d'instructions d'un centre de commande central. Néanmoins, le contrôle décentralisé ne signifie pas que les robots n'ont pas besoin de communiquer entre eux. Chaque robot est censé recevoir des signaux pour savoir à tout moment quelle est la meilleure solution globale et où. De même, un robot qui trouve une meilleure solution est censé transmettre les informations sur les nouvelles meilleures solutions mondiales aux autres robots.
- *Détection de température :* le capteur de température utilisé est Micro chip MCP9700A. Le potentiomètre sur le circuit de conditionnement du signal permet à l'utilisateur d'ajuster la plage du signal pour différentes températures ambiantes. Il permet également à l'utilisateur d'étalonner les différents robots pour permettre des lectures cohérentes sur les trois robots.

La mise en œuvre de l'algorithme a présenté des défis multiples qui ne seraient pas observés dans une implémentation informatique. Ces défis sont : les capacités de calcul des robots, navigation hors limites, collision, détection de la cohérence (sensing consistency). Les résultats expérimentaux montrent que l'algorithme modifié a toujours été en mesure de trouver le point maximum global en pas plus de 80 secondes, il a également montré que le changement de la taille d'incrément des robots n'avait pas d'effet significatif sur l'efficacité de l'essaim ou sa convergence.

5) Using auction-based task allocation scheme for simulation optimization of search and rescue in disaster relief (Tang, et al., 2018)

Tang & al. (Tang, et al., 2018) Ont proposé d'optimiser la recherche et le sauvetage grâce à une simulation basée agents qui utilise la marche aléatoire Lévy.

Les auteurs simulent le cas catastrophe avec le cas du sauvetage par glissement de terrain. Ils construisent un environnement post-catastrophe basé sur une modélisation Multi-Agents. Les probabilités de survie des victimes prises au piège dans les décombres devraient se détériorer avec le temps. Lié à la recherche de victimes, l'environnement est une zone entièrement recouverte de gravats, de roches, de débris, etc. Il est difficile d'obtenir des informations critiques sur l'environnement (telles que l'emplacement des victimes). Ils adoptent les « Truncated Lévy walks » pour simuler les comportements de recherche des équipes de secours.

Les agents actifs peuvent réagir à l'environnement et agir sur lui. Les tâches sont réparties de manière aléatoire dans une zone et diffèrent par certaines propriétés, telles que la priorité des tâches, les charges de travail requises pour les terminer. Deux types d'agents actifs sont inclus, à savoir les victimes et les équipes de secours. Les paramètres essentiels des victimes comprennent la profondeur enfouie, les coordonnées et la gravité des blessures :

- La profondeur enfouie indique la quantité de gravats sur les victimes piégées. On suppose que les victimes dans le même site enterré sont égales en profondeur enterrée contre les gravats.
- Les coordonnées indiquent l'emplacement des victimes.
- Gravité des blessures, trois scénarios sont supposés (« Fatal », « Serious » et « Normal »), et ils seront combinés avec la gravité des blessures.

Tandis que les paramètres essentiels des équipes de sauvetage comprennent le rayon de recherche (l'étendue maximale de recherche de survivants), l'angle de braquage maximal (la plage d'angle de braquage à chaque étape), la vitesse de sauvetage (la vitesse de déplacement des décombres par une équipe de sauvetage), l'étendue de la coopération (la coopération se produit uniquement entre les équipes de secours dans cette zone spécifiée et qu'elle est limitée par la portée de communication), la longueur du mouvement, l'angle de braquage(sont tirés de distributions de probabilités particulières. Les étapes sont terminées lorsqu'une longueur de déplacement ou une limite est atteinte, ou lorsqu'un site enterré est détecté. À la fin d'une étape, les équipes de secours se déplacent vers le site enfoui ou sélectionnent l'étape suivante pour continuer la recherche).

L'algorithme d'attribution des tâches basé sur les enchères a été utilisé pour répartir la tâche de sauvetage entre les équipes de sauvetage dans le cadre de la coopération. Il existe deux types d'éditeur de tâche et de répondeur de tâche, correspondant respectivement aux commissaires-priseurs (Auctioneers) ; Les équipes de secours qui ont détecté un site enterré dans le rayon de recherche et aux enchérisseurs (Bidders) ; Les

équipes de secours qui ont reçu des messages d'enchères jouent le rôle de soumissionnaires. Les soumissionnaires ne soumissionneraient que pour une seule tâche à chaque fois. Lors de la réception de plusieurs messages d'enchères, les soumissionnaires sont censés choisir la tâche qui pourrait générer l'utilité la plus nette pour soumissionner. Le but de l'algorithme basé sur les enchères est la maximisation des performances globales.

Les auteurs ont comparé le plan de sauvetage, proposé au plan de sauvetage non coopératif (dans le plan de sauvetage non coopératif, les équipes de sauvetage ne sauvent que les victimes qui sont détectées par elles-mêmes), puis le comparons avec le plan de sauvetage basé sur F-Max-Sum (il s'agit d'un algorithme décentralisé, qui est développé sur la base de Max-Sum). La simulation est implémentée dans NetLogo et R. Les paramètres de cette simulation ont les caractéristiques de nombreux cas du monde réel. La zone affectée est caractérisée comme une zone circulaire de rayon 100 m. En supposant qu'il y a 100 victimes, 20 équipes de secours dans ce domaine. Les victimes se trouvent dans des sites enterrés répartis au hasard. L'unité des étapes de simulation est la minute. Un cycle de simulation se poursuivrait jusqu'à ce que tous les survivants soient secourus, et le délai d'un cycle de simulation est fixé à 72 h. les simulations ont été répété 200 fois pour obtenir des résultats de simulation de haute précision.

Les résultats de la simulation indiquent que le plan de sauvetage coopératif pourrait améliorer considérablement l'efficacité du sauvetage, et il fonctionne un peu mieux que l'approche basée sur F-Max-Sum en ce qui concerne certains indicateurs. En outre, sa faible complexité le rend plus approprié pour la coopération entre les équipes de secours que F-Max-Sum. L'analyse de robustesse montre que le rayon de recherche peut affecter considérablement l'efficacité du sauvetage, tandis que la portée de la coopération a peu d'effet sur l'efficacité du sauvetage. L'analyse de sensibilité montre que les deux paramètres, le délai pour terminer les opérations de sauvetage dans un site enterré et l'angle de braquage maximal pour l'étape suivante, ont tous deux une grande influence sur l'efficacité du sauvetage, et il existe une valeur optimale pour les deux en vue d’efficacité du sauvetage.

Le Tableau 2.1 montre une comparaison des travaux relies selon certains critères que nous jugeons importants.

Axe 1	Axe 2	Axe 3	(Abuomar & Al-Aubidy, 2018)	**(Kumar et al.,2017)**	**(Greenhagen et al., 2016)**	**(Din et al., 2018)**	(Tang, et al., 2018)
Environnement	Complexité	Avec obstacle	X	X		X	X
		Sans obstacle			X		
Objets	Distribution	Aléatoire	X	X	X	X	X
		Partiellement regroupés					
		Entièrement regroupés					
	Nature	Statique	X	X	X	X	X
		Dynamique					
Dépôts	nombre	Seul	X	X	X	X	
		Multiple					
Robots	énergie	Limitée					
		Illimitée	X	X	X	X	X
	perception	Illimitée					
		Limité	X	X		X	
	mémorisation	Oui			X	X	X

		Non	X	X			
Stratégies	communication	Directe	X			X	
		Indirecte		X	X		X
		Inspiration				VL	
	exploration	Aléatoire		X			
		Stratégique	X		X	X	X
		Inspiration	BDA	RDPSO	PSO	VL	Lévy
	recrutement	Oui		X		X	
		Non	X		X		
		Inspiration				VL	
Simulation	Monde réelle	Expérimentation réel			X		
	Plateforme	Argos				X	
		Player/ Stage					
		Webots					
		Gazebo		X			
		auto développé	X				X

Tableau 2.1 : Comparaison qualitative des travaux reliés.

Concernant le **Tableau 2.1**, nous concluons que :

- Toutes les expériences utilisées dans un environnement avec des obstacles ont été distribué aléatoirement.
- Toutes les expériences ont utilisé un seul dépôt et la distribution des cibles était aléatoire.
- Ils ont utilisé des algorithmes bio-inspirés comme stratégies d'exploration et des comportements de communication directe et indirecte pour les robots.
- Les expériences ont été testé soit sur ARGoS, soit sur une plateforme auto-développée.

L'objectif principal de ce travail est de localiser les victimes et de les transporter vers un dépôt central. La comparaison des travaux dans le tableau ci-dessus montre déjà que les expériences qui ont été faites se sont concentrées uniquement sur la recherche et négligent le retour au dépôt.

IV Conclusion

Dans ce chapitre, nous avons défini l'application de recherche et sauvetage Multi-Robot. Ensuite, nous avons présenté et discuté des travaux connexes disponibles dans ce domaine. Nous avons terminé cette partie par un tableau de comparaison qui nous a permis de faire plusieurs observations importantes pour proposer une solution adéquate.

Chapitre 3 : Conception

I Introduction

Dans la foraging, les robots sont chargés de chercher, collecter et transporter un ou plusieurs objets vers un ou plusieurs dépôts. La recherche et le sauvetage est considérée comme étant l'application du foraging dans la vie réelle, donc une telle application partage les même caractéristiques et définitions que celle de foraging.

Dans ce travail, nous avons proposé une solution au problème de recherche et la sauvegarde des victimes et nous avons utilisé un modèle d'interaction robotique basé sur les algorithmes Dragon Fly et Lévy-walk.

Nous commençons ce chapitre par présentation de la problématique. Nous présentons ensuite l'objectif à atteindre ainsi qu'une présentation détaillée de la solution que nous avons proposée. Ensuite, nous nous concentrons sur l'algorithme proposé, le comportement de nos robots à travers une machine d'états et les pseudo codes des différents comportements. Nous terminons le chapitre par une conclusion.

II Problématique, objectifs et proposition

La recherche et le sauvetage est reconnue comme une instance du foraging dans le monde réel. Elle consiste à rechercher des victimes dans des zones complexes et inaccessibles aux humains. Comme le foraging, une telle application consiste en deux étapes importantes : (1) l'exploration, et (2) le retour au dépôt. Pour avoir une stratégie de sauvetage (voir du foraging) efficace, les deux comportements précédents doivent prendre le minimum de temps possible. L'exploration pourrait être efficace si les robots s'éloignent les uns des autres et couvrent le maximum possible des zones et le retour au dépôt pourrait être efficace et rapide si les robots sont guidés au dépôt sans aucune communication directe et si elle n'est pas couteuse.

Dans ce travail nous cherchons à proposer des comportements efficaces d'exploration et de retour au dépôt dans l'objectif à chercher et sauver le maximum possible de victimes dans des temps minimaux.

Nous avons fixé deux objectifs principaux :

1. Assurer une large dispersion des robots pour couvrir le maximum possible de zones et de détecter le maximum possible de victimes ;
2. Guider les robots après une détection de victime pour se rendre rapidement au dépôt ;
3. Eviter les communication directe ou indirecte entre les robots pour assurer une solution non couteuse ;

Pour répondre aux objectifs fixés précédemment, nous avons jugé intéressant d'utiliser des algorithmes issus de l'intelligence en essaim. Premièrement, nous avons utilisé le dragon Fly basée sur la marche aléatoire Levy, dans cette marche les robots utilisent une équation qui calcule la prochaine position, ce qui permet une dispersion dans la totalité de l'espace de recherche. Deuxièmement, nous avons utilisé des spots lumineux dans la base et dans l'environnement qui permettent de guider les robots au dépôt. Les robots commencent par une marche aléatoire, jusqu'à atteindre une lumière, dans ce cas ils suivent la lumière jusqu'au dépôt.

III Algorithmes reliés

Les algorithmes que nous avons utilisés dans notre programme sont : Lévy Walk et l'algorithme de libellule. Nous donnerons dans cette section, une description plus au moins détaillée de ces deux algorithmes :

IV.1. DragonFly Algorithme

L'algorithme Dragonfly (DA) a été développé par Mirjalili en 2016, est un algorithme d'optimisation méta-heuristique récent et intéressant inspiré de la nature, utilisé pour résoudre une grande variété de problèmes d'optimisation. Il existe 3000 espèces différentes de libellules et leur cycle de vie comprend deux étapes appelées nymphe et adulte.

Les essaims dynamiques (migratoire) et statiques (alimentation) constituent respectivement les phases d'exploitation et d'exploration de l'DA. Où Dans la phase d'exploitation, un grand nombre de libellules font migrer les essaims dans une direction sur de longues distances et distraient les ennemis. Et dans la phase d'exploration, cependant, les libellules forment de petits groupes et volent dans une petite zone pour chercher de la nourriture et attirer des proies volantes.

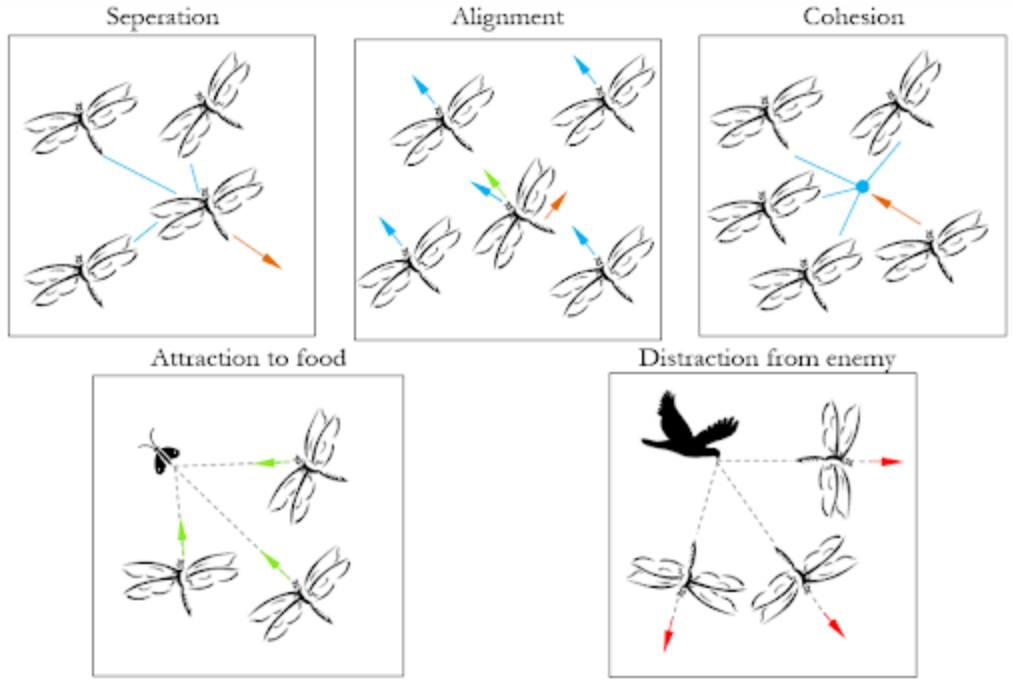


Figure 3.1 : Primitive corrective patterns between individual in swarm (Palappan & Thangavelu, 2018)

Cinq primitifs de base sont utilisés pour modéliser les comportements des essaims de libellules (où p représente la position de l'individu courant, p_j la position du j ème individu voisin et m le nombre d'individus voisins) :

1) **Séparation :** représente l'évitement de collision statique que les individus suivent pour éviter une collision avec d'autres individus dans le quartier ;

$$\mathbf{S_i} = -\sum_{j-1}^{m} \boldsymbol{p} - \boldsymbol{p_j} \quad \text{.....(1)}$$

2) **Alignement :** indique l'appariement de la vitesse de l'individu entre d'autres individus du quartier du même groupe ; (où v_j désigne la vitesse du j ème individu).

$$\boldsymbol{A_i} = \frac{\sum_{j-1}^{m} \boldsymbol{v_j}}{\boldsymbol{m}} \quad \text{.....(2)}$$

3) **Cohésion :** représente la tendance des individus vers le centre du groupe d'essaims ;

$$C_i = \frac{\sum_{j-1}^{m} p_j}{m} - p \quad \text{.....(3)}$$

4) **Attraction :** (où F_i représente la source de nourriture du i ème individu et Fp est la position de la source de nourriture) ;

$$F_i = Fp - p \quad \text{.....(4)}$$

5) **Distraction :** (où E_i désigne la position de l'ennemi du i ème individu et E_p désigne la position de l'ennemi) ;

$$E_i = E_p + p \quad (5)$$

L'algorithme DA est développé sur la base du cadre de l'algorithme PSO où le vecteur pas à pas montre la direction du mouvement du dragon et défini comme suit (où ΔX_t représente le vecteur de pas et t l'itération actuelle, et s, a, c, f, e, w sont des poids de séparation, alignement, cohésion, nourriture, ennemi, et inertie):

$$\Delta X_{t+1} = (s\mathbf{S_i} + a\boldsymbol{A_i} + cC_i + fF_i + eE_i) + w\Delta X_t \quad (6)$$

Après avoir calculé le vecteur de pas, le vecteur de position (X_t) est calculé comme suit :

$$X_{t+1} = X_t + \Delta X_{t+1} \quad (7)$$

IV.2. Levy-walk :

La marche de Lévy est une stratégie de recherche aléatoire, la trajectoire du processus de Lévy est caractérisée par un ensemble de petits pas reliés par un long pas, peut être approximée par la fonction de densité dans l'équations (Où U est un nombre flottant distribué dans un intervalle de [0,1] et l_0 est la longueur initiale) :

$$P(x) \sim |x[- 1 - \beta \ avec \ (0 < \beta \leq 2 \ et \ x \longrightarrow \infty \quad (8)$$

$$P(l) = \frac{\beta}{L_0 \left(1+\frac{l}{l_0}\right)^{1+\beta}} \quad (9)$$

$$l = l_0 \left(\frac{1}{U^{1/\beta}} - 1\right) \quad (10)$$

IV Algorithme proposé

Nous présentons dans cette partie, une description de l'algorithme propose, la machine d'état et le pseudo code.

IV.1. Description

- ***Cherche :*** nous avons utilisé l'équation DragonFly-Levy (Palappan & Thangavelu, 2018) pour permettre aux robots de chercher et sauvetage :

$$X_{t+1} = X_t + Levy(d) \times X_t \quad (11)$$

$$Levy(d) = 0.01 \times \frac{r_1 \times \sigma}{|r_2|^{\frac{1}{d}}} \quad (12)$$

$$\sigma = \left(\frac{\tau(1+\beta)\times\sin(\pi\beta/2)}{\tau((1+\beta)/2)\times\beta\times 2^{((\beta-1)/2)}}\right)^{1/\beta} \quad \text{(13)}$$

$$\tau(x) = (x-1)! \quad \text{(14)}$$

Deux cas se présentent :

1. Si le robot détecte une victime et le nombre de victimes = N (où N est prédéfini et représente le nombre de victime maximum qui peut être transporté par le robot en même temps; il passe à l'état ***sauvetage***.
2. Si le robot détecte un autre robot, un obstacle ou atteint l'un des dépôts il passe à l'état ***Eviter_obstacle***.

- ***Sauvetage :*** le robot cherche les lumières et les suivent. On a deux conditions qui se présentent aussi :
  1. Si le robot est dans l'un des dépôts ; il passe à ***Déposer_Victime*** Sinon il continu à suivre la lumière.
  2. Si le robot détecte un autre robot ou un obstacle dans l'environnement ; il passe à l'état ***Eviter_obstacle***.
- ***Déposer_Victime :*** Si le robot est dans le dépôt ; il dépose tous les victimes et passe directement à l'état ***Cherche.***
- ***Eviter_obstacle :*** le robot calcule avec ses 24 capteurs la valeur de proximité pour détecter les obstacles et les robots dans son voisinage. S'il y a un obstacle proche ou des robots, il tourne dans la direction opposée, sinon le robot continu l'exécution de son état courant : l'état ***sauvetage*** ou l'état de ***cherche***.

IV.1. La Machine d'état de notre algorithme

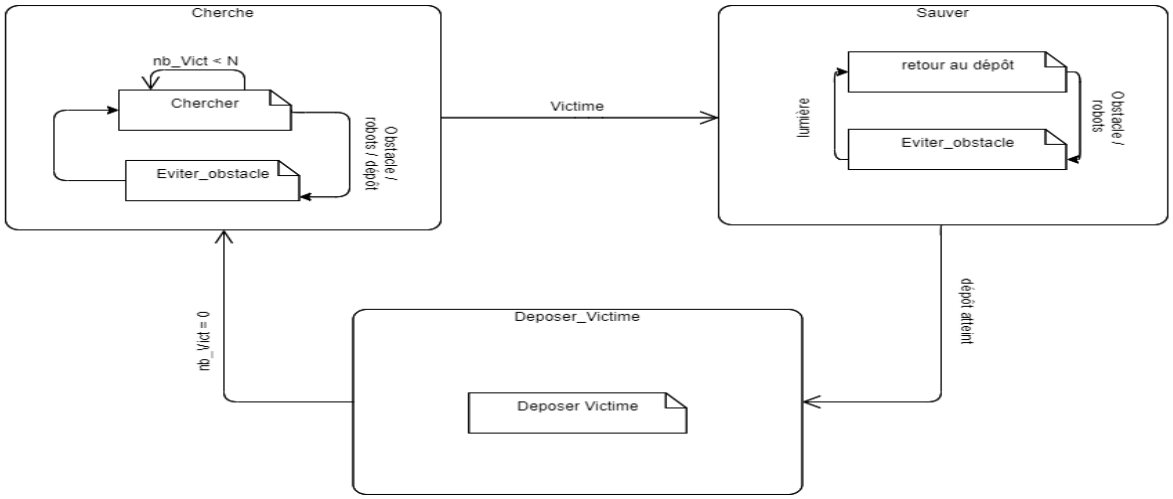


Figure 3.1: machine d'état de l'algorithme DAL

IV.2. Pseudo code

Algorithme III. 1 : Chercher

1. **Tant que** ∄(obstacle ou robot ou dépôt) et (nb_Vict<N) **faire**
2. Marche aléatoirement en utilisant l'équation 11
3. **Fin Tant que**
4. **Si** ∃**(obstacle ou robot**) **alors Aller à** Eviter_obstacle
5. **Sinon Si** (∃ **dépôt**) **alors Aller on** arrière
6. **Sinon Si (nb_Vict<N) alors Aller** à Sauver
7. **Fin Si**

Algorithme III. 2 : Sauver (retour au dépôt)

1. **Tant que** ∄(dépôt ou obstacle ou robot) **faire**
2. Suivie lumière des dépôts
3. **Fin Tant que**
4. **Si** ∃ **(obstacle ou robot**) **alors Aller à** Eviter_obstacle
5. **Sinon Si** (∃ **dépôt**) **alors Aller à** Déposer_Victime
6. **Fin Si**

Algorithme III. 3 : Déposer_Victime

1. **Si** le robot est dans le dépôt **alors** il dépose les victimes et passe à l'état **chercher**
2. **Fin Si**

Algorithme III. 4 : Eviter_obstacle

1. **Récupérer** les lectures de tous les capteurs de proximité V_i
2. **Si** ($V_i > 0$) **alors**
3. **Calculer** l'angle de V_i
4. **Si** A montre que l'obstacle est devant le robot **alors** Marche en arrière
5. **Sinon Si** A montre que l'obstacle est dans le côté gauche **alors** Déplacer à droite
6. **Sinon Si** A montre que l'obstacle est dans le côté droite **alors** Déplacer à gauche
7. **Fin Si**
8. **Sinon Si (état Cherche) alors Aller à** Chercher
9. **Sinon Si (état Sauver) alors Aller** à Sauver
10. **Fin Si**
11. **Fin Si**

II Conclusion

Nous avons proposé dans ce chapitre un algorithme de recherche et de sauvetage (voir du foraging) qui hybride deux méthodes bio-inspirées : le Lévy walk et le dragonfly. L'idée été d'assurer une large dispersion des robots avec le Lévy walk, puis de les guider au dépôt pour se rendre rapidement au dépôt avec les spots lumineux.

Nous avons commencé par présenté la problématique, les objectifs et proposition, puis nous avons expliqué brièvement les deux algorithmes reliés. Par la suite, nous avons montré notre proposition, sa machine d'état et les pseudos-codes associés.

Chapitre 4 : Implémentation et simulations

I Introduction

Dans ce chapitre, nous testons les performances et la validation de l'algorithme proposé DragonFly Avec Levy (DAL). Nous l'avons implémenté sous la plateforme robotique ARGoS. Pour tester l'impact des améliorations apportées, nous avons effectué plusieurs simulations dans différentes configurations environnementales et pour comparer les résultats avec d'autres algorithmes, nous avons implémenté un algorithme aléatoire simple.

Nous commençons par présenter la plateforme de simulation ARGoS. Puis nous expliquons les caractéristiques des robots utilisés. Nous proposons ensuite un ensemble de scénarios de simulation. Ensuite, nous discutons et comparons les résultats obtenus, et nous terminons par une conclusion.

II Environnement de développement

ARGoS est un simulateur multi-robot open source qui a d'abord été développé dans le cadre du projet Swarmanoid financé par l'UE pour étudier les outils et les stratégies de contrôle des essaims hétérogènes de robots. Aujourd'hui, il s'agit d'une plateforme de simulation largement utilisée dans de nombreuses recherches et projets dédiés aux contrôles de synthèse pour les comportements d'essaimage.

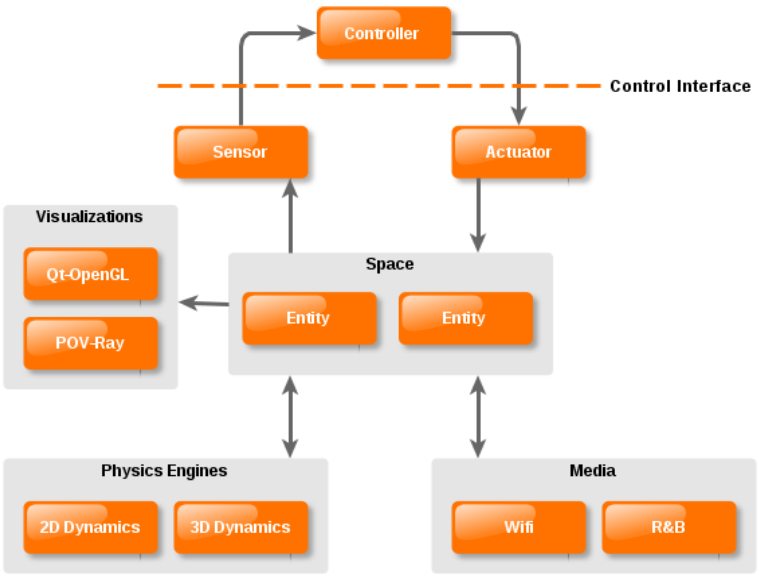


Figure 4.1 : l'architecture du simulateur ARGoS

ARGoS est livré avec tous les outils nécessaires au cycle de développement du code de contrôle des robots, de la conception à la validation sur de vrais robots. Par conséquent, il n'y a aucune différence entre le codage pour la simulation ou la réalité. ARGoS peut simuler à grande échelle des essaims hétérogènes de robots en simulation en temps réel. Son architecture est conçue pour être flexible de telle sorte que les fonctionnalités personnalisées peuvent être facilement modifiées ou ajoutées sous forme de plug-ins ou de modules.

III Modélisation des composants de notre système

Notre système se compose de quatre composants principaux : l'environnement (un espace 3D continu), les Obstacles (des différentes formes de boîtes grises 3D de tailles différentes), les victimes (des cylindres grises 3D) et pour les robots on utilise les foot-bot. La Figure 4.2 montre une capture de DAL sous ARGoS.

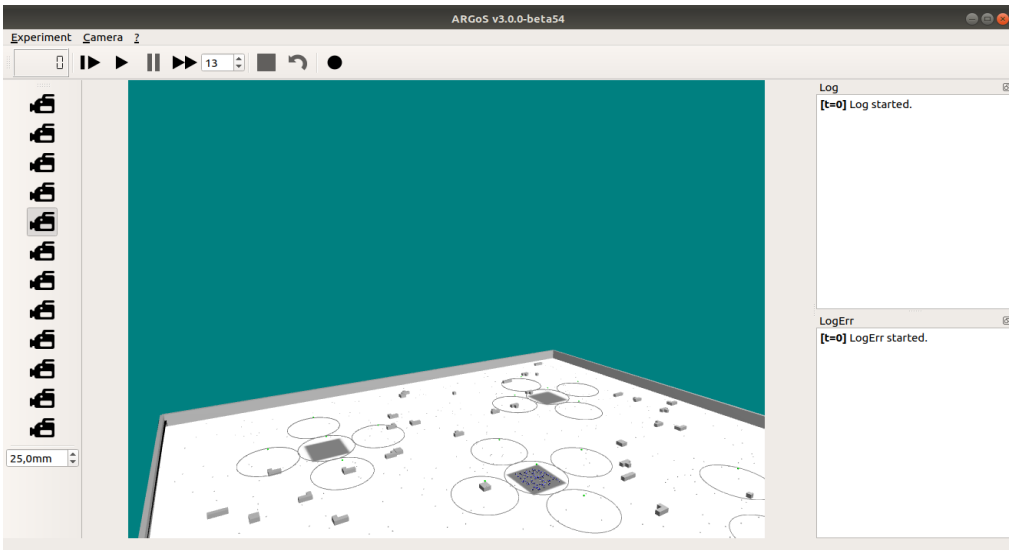


Figure 4.2 : Capture de DAL sous ARGoS

III.1 Foot-bot

Le foot-bot est un robot mobile différentiel à deux roues d'environ 17 cm de diamètre et 29 cm de hauteur, conçu et construit dans le cadre du projet SWARMBOTS. Le pied-bot peut se déplacer en utilisant une combinaison de roues et de pistes (appelées arbres), et il est livré avec divers capteurs et actionneurs qui permettent une interaction avec l'environnement environnant. Par la suite, une liste des plus pertinentes utilisées dans les études de robotique en essaim :

- Douze (12) LED RVB composant un anneau qui entoure le corps du robot, et à travers des motifs colorés de sorcière peuvent être affichés aux autres robots.

- Une caméra omnidirectionnelle qui peut être utilisée pour percevoir des objets colorés affichés par d'autres robots (jusqu'à une distance d'environ 50 cm).
- Quatre (4) capteurs au sol placés sous le châssis peuvent être utilisés pour percevoir des repères ou des trous sur le sol.
- Vingt-quatre (24) capteurs IR pour la détection d'obstacles et de proximité.
- Un dispositif de communication à distance et relèvement, appelé RAB, pour échanger des messages entre robots dans une portée limitée.
- Un connecteur de préhension pour permettre au foot-bot d'effectuer des connexions physiques telles que la prise d'abjects ou des connexions robot-robot.

IV Simulation et analyse des résultats

Dans cette section, nous présentons les critères de performances utilises, les scenarios de simulation réalisés, les résultats obtenus et les discussions.

IV.1 Critères de performances

Nous avons utilisé un seul critère de performance : le pourcentage de sauvetage dans un temps fixe (30 minutes) dans des configurations environnementales différentes (voir scenarios 1, 2, 3, 4 et 5).

IV.2 Scénarios de simulation

Les cinq scénarios de simulation (qui sont présentent dans le Tableau 4.1.) testent l'effet de la variation du nombre de robots, de la taille de l'environnement, du nombre de victimes, de la distribution des victimes et du nombre de victimes sauver sur les performances de l'algorithme proposés DAL.

Scénarios 1 : variation du nombre de robots	**Scénarios 2 : variation de la taille de l'environnement**
Nombre des robots : 50, 70, 100, 120 ; Distribution des victimes : Aléatoire ; Nombre des victimes : 500 ; Densité des obstacles : 13% ; Taille de l'environnement : 120m X 120m ;	Nombre des robots : 70 ; Distribution des victimes : Aléatoire ; Nombre des victimes : 500 ; Densité des obstacles : 13% ; Taille de l'environnement : 80 m X 80 m, 100 m X 100 m, 140 m X 140 m, 200 m X 200 m

Scénarios 3 : variation du nombre de Victimes	**Scénarios 4 : la distribution des victimes**
Nombre des robots : 70 ; Distribution des victimes : Aléatoire ; Nombre des victimes : 500, 600, 800, 1000 ; Densité des obstacles : 13% ; Taille de l'environnement : 120m X 120m ;	Nombre des robots : 70 ; Distribution des victimes : Aléatoire, cluster ; Nombre des victimes : 500 ; Densité des obstacles : 13% ; Taille de l'environnement : 120m X 120m ;
Scénarios 5 : variation du nombre de victimes sauver	
Nombre des robots : 70 ; Distribution des victimes : Aléatoire ; Nombre des victimes : 500 ; Nombre des victimes sauver : 1, 2, 4, 5 ; Densité des obstacles : 13% ; Taille de l'environnement : 120m X 120m ;	

Tableau 4.1 : Scenarios de simulations réalisés

IV.3 Résultats et discussions

Dans ce qui suit, nous présenterons les résultats obtenus par le DLA et l'algorithme aléatoire dans différents scénarios, les résultats seront discutés et analysés (Les résultats obtenus dans les expériences suivantes sont la moyenne de 5 simulations dans 30 min).

1) Scénario 1

Dans ce scénario, Nous étudions l'influence du nombre de robots sur les performances de l'algorithme.

N° Robots / Algorithmes	50	70	100	120
DAL	62.4%	68.4%	70.8%	74.2%
Random	24.4%	31.6%	19%	17.6%

Tableau 4.2 : le premier scenario : variation du nombre de robots

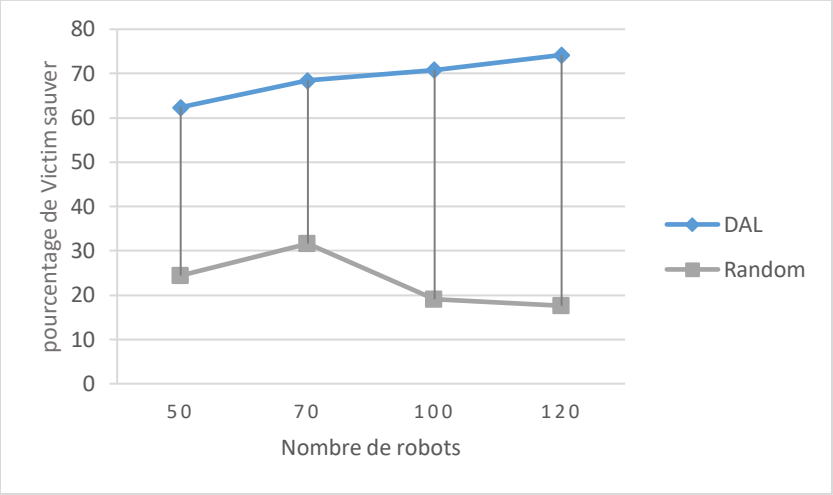


Figure 4.3 : le premier scenario : variation du nombre de robots

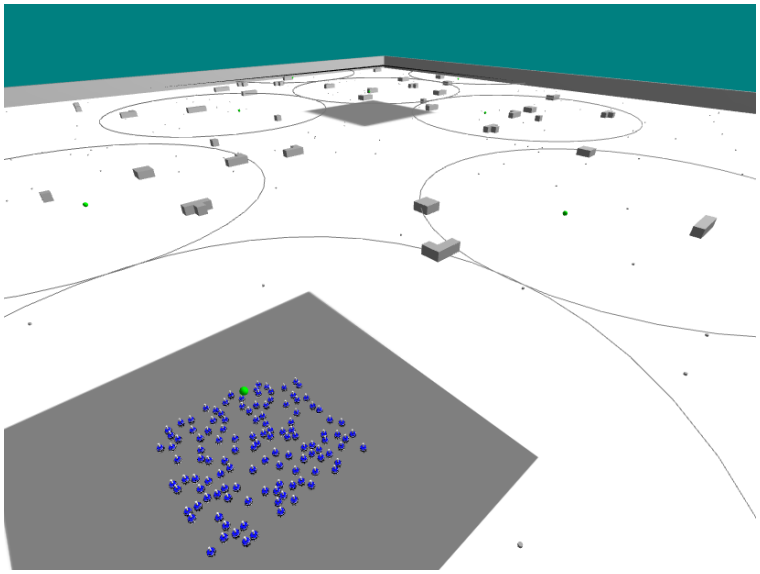


Figure 4.4 : Une exécution sous ARGoS de DAL avec 120 robots

Discussion et comparaison

Le Tableau 4.2 et la Figure 4.3 montrent les résultats obtenus avec les algorithmes DAL et Random. Les résultats montrent que l'augmentation du nombre de robots conduit à une augmentation du pourcentage de sauvetage avec l'algorithme DAL. Dans Random, il y a eu une augmentation lorsque nous avons utilisé 70 robots et ensuite le graphique s'est dégradé ; tandis qu'en DAL il y avait une augmentation significative du nombre des victimes sauver par les robots. Cependant, les résultats obtenus en DAL sont meilleurs par rapport au Random.

2) Scénario 2

Dans ce scénario, Nous étudions l'influence de la taille de l'environnement sur les performances de l'algorithme.

Taille / Algorithmes	80x80	100x100	120x120	160x160
DAL	99.9%	86.4%	62.4%	25%
Random	45%	39.8%	24.4%	8%

Tableau 4.3 : le deuxième scenario : variation de la taille de l'environnement

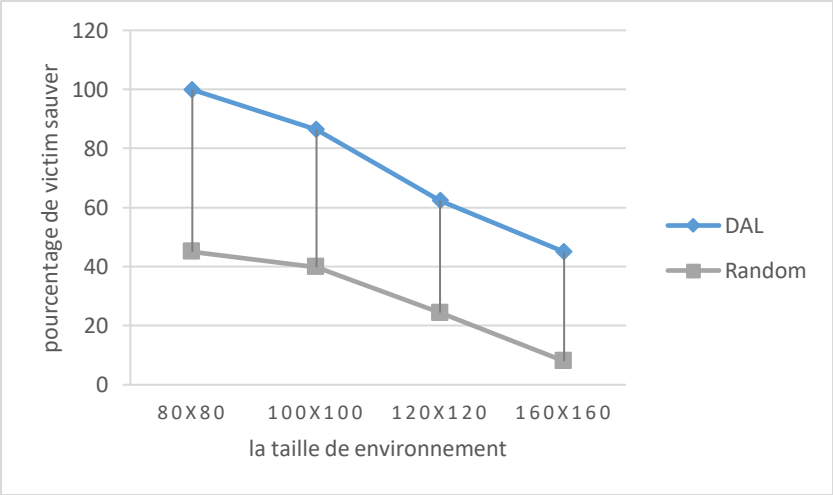


Figure 4.5 : le deuxième scenario : variation de la taille de l'environnement

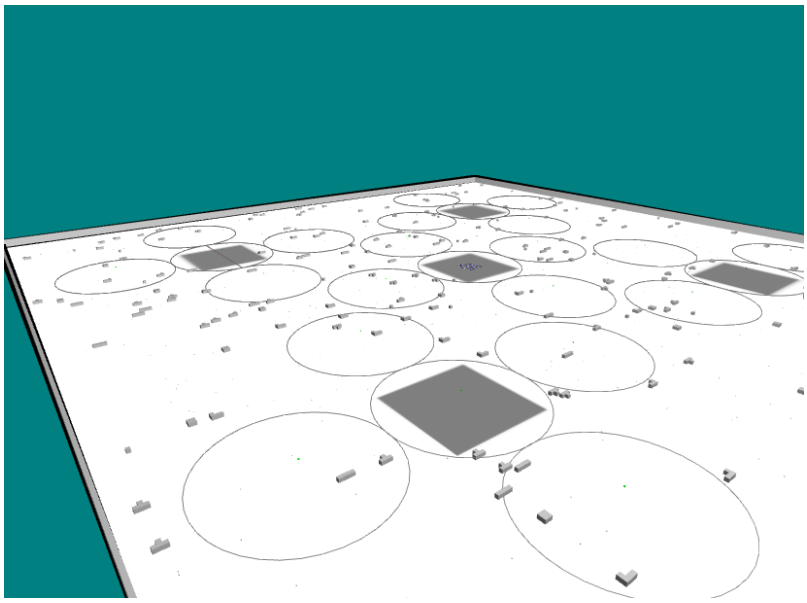


Figure 4.6 : Une exécution sous ARGoS de DAL avec taille de 160 x 160

Discussion et comparaison

Le Tableau 4.3 et la Figure 4.5 montrent les résultats obtenus avec les algorithmes DAL et Random en changent la taille de l'environnement. Les résultats montrent que l'augmentation la taille de l'environnement conduit à une décrémentation du pourcentage de sauvetage dans les deux algorithmes. Cependant, les résultats obtenus en DAL sont meilleurs que ceux de Random.

3) Scénario 3

Dans ce scénario, nous étudions l'influence de nombre de victimes sur les performances de l'algorithme.

N° victimes / Algorithmes	500	600	800	1000
DAL	62.4%	58.17%	41.13%	34.8%
Random	24.4%	21.17%	27.58%	17.77%

Tableau 4.4 : le troisième scenario : variation du nombre de Victimes

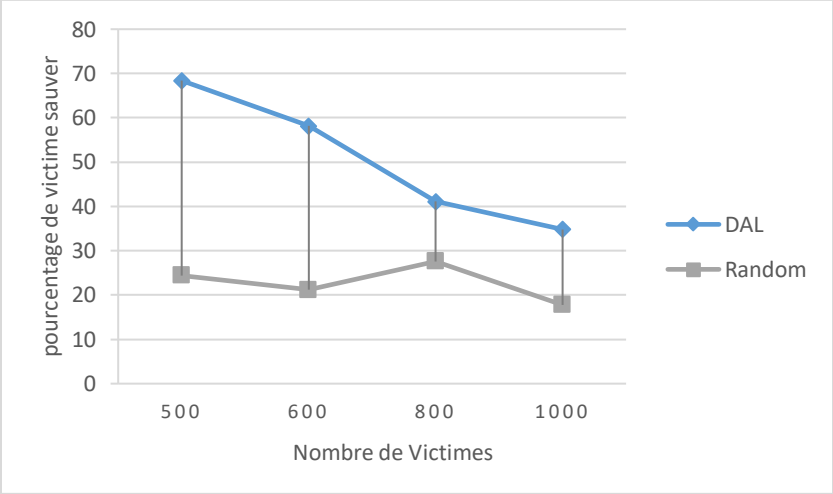


Figure 4.7 : le troisième scenario : variation du nombre de Victimes

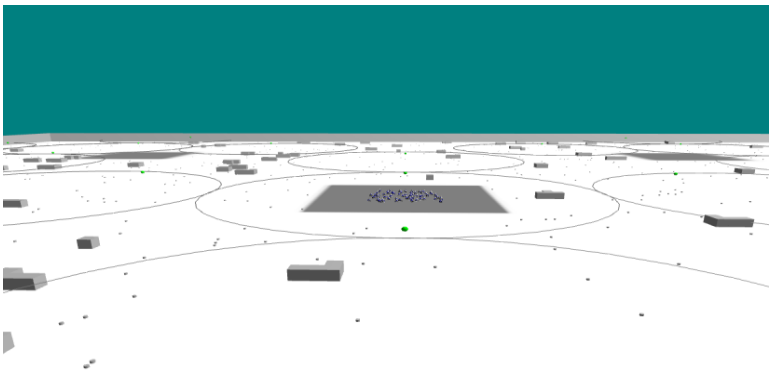


Figure 4.8 : Une exécution sous ARGoS de DAL avec 1000 victime

Discussion et comparaison

Le Tableau 4.4 et la Figure 4.7 montrent les résultats obtenus avec les algorithmes DAL et Random en changent le nombre de victimes. Les résultats montrent que l'augmentation de nombre de victime conduit à une décrémentation du pourcentage de sauvetage dans l'algorithme DAL. Avec Random, il y a eu une décrémentation, mais il y a eu une incrémentation significative quand le nombre de victimes est 800. Cependant, les résultats obtenus avec DAL sont meilleurs que ceux de Random.

4) Scénario 4

Victimes / Algorithmes	Aléatoire	Cluster
DAL	62.4%	17.6%
Random	24.4%	10.67%

Tableau 4.5 : le quatrième scenario : la distribution des victimes

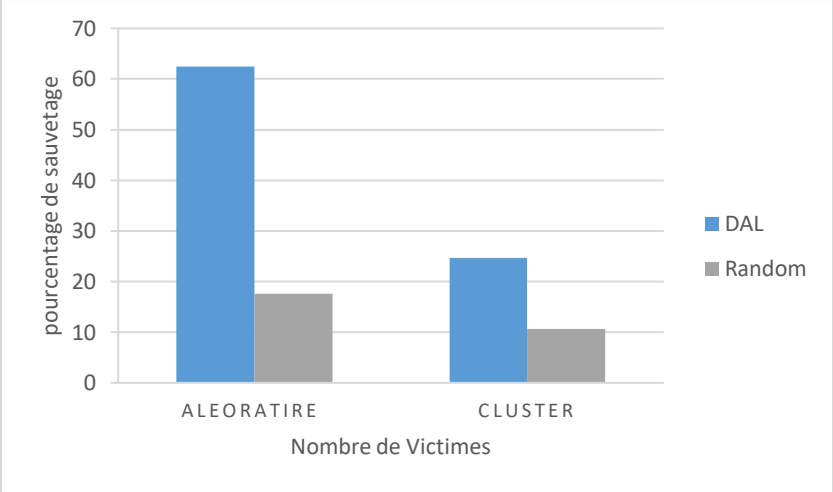
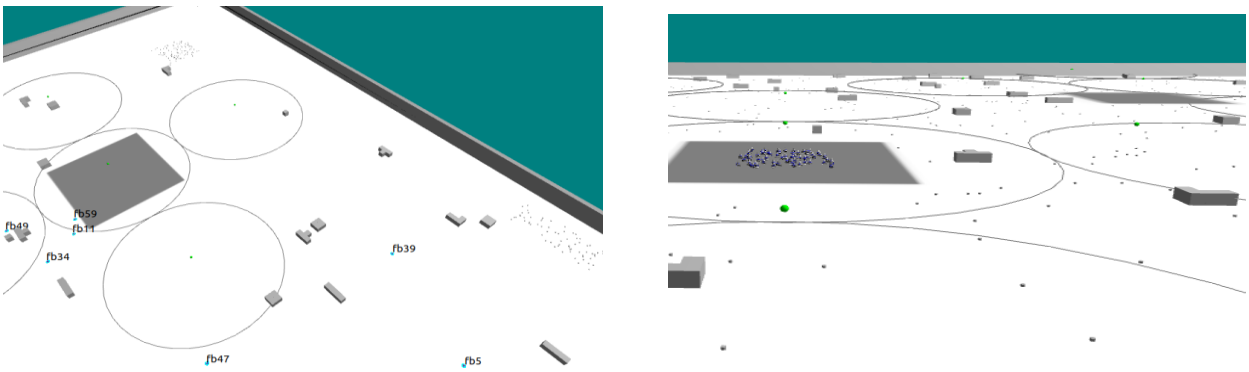


Figure 4.9 : le quatrième scenario : la distribution des nombre de victimes sauver

Discussion et comparaison

Le Tableau 4.5 et la Figure 4.9 montrent les résultats obtenus avec les algorithmes DAL et Random en modifiant la distribution des victimes. Les résultats montrent que lorsque la distribution est aléatoire le pourcentage de sauvetage est meilleur que lorsque la distribution est en clusters. Cependant, les résultats obtenus en DAL sont meilleurs que ceux de Random.



(a) : cluster **(b) :** aléatoire

Figure 4.10 : Une exécution sous ARGoS de DAL avec diffèrent distribution des victime

5) Scénario 5

Dans ce scénario, Nous étudions l'influence de nombre de victimes qui on peut sauver sur les performances de l'algorithme (Les résultats obtenus dans les expériences suivantes sont dans un temps fixe de 20 min).

N° sauver / Algorithmes	1	2	4	5
DAL	62.4%	90.36%	70.44%	85.68%
Random	24.4%	60%	69.87%	58.67%

Tableau 4.6 : le cinquième scenario : variation du nombre de victimes sauver

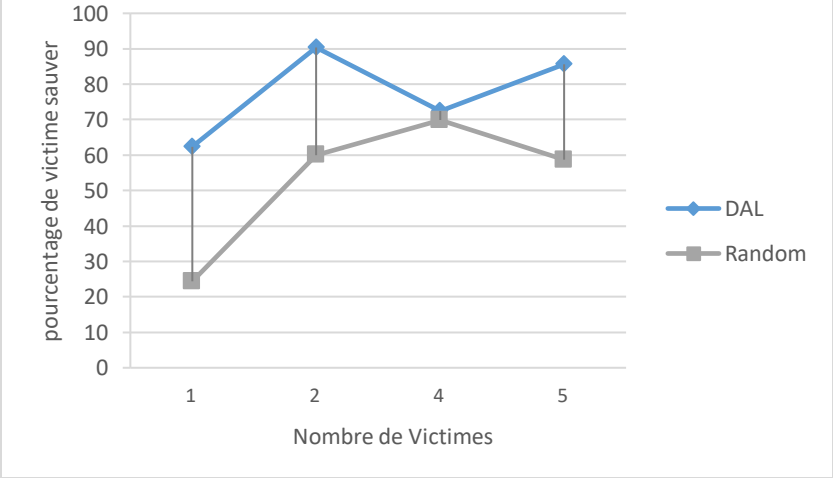
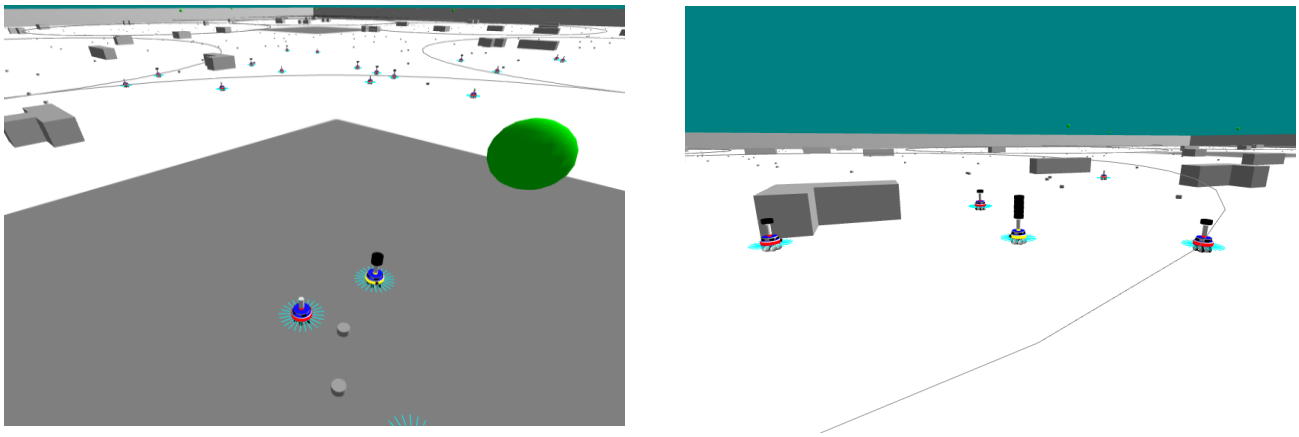


Figure 4.11 : le cinquième scenario : variation du nombre de victimes sauver



(a) : 2 victime **(b) :** 5 victime

Figure 4.12 : Une exécution sous ARGoS de DAL changer le nombre de victime possible à sauver

Discussion et comparaison

Le Tableau 4.6 et la Figure 4.11 montrent les résultats obtenus avec les algorithmes DAL et Random en modifiant le nombre de victimes à transporter à la fois. Les résultats montrent qu'une augmentation du nombre de victimes possibles à sauver conduit à une augmentation dans le pourcentage de sauvetage de l'algorithme DAL. Avec Random, le plus grand nombre possible de victimes à sauver conduit à une augmentation des résultats mais lorsque nous arrivons à la quantité 5 victimes, il a commencé à diminuer, tandis que dans l'algorithme DAL les résultats commencent par augmenter puis en 4 ils diminuent et recommencent à augmenter en 5 victimes. Cependant, les résultats obtenus en DAL sont meilleurs que ceux de Random.

V Conclusion

Dans ce chapitre, nous avons présenté la plateforme ARGoS utilisée et le robot Footbot, ses caractéristiques et ses composants, puis nous avons proposé un ensemble de scénarios avec variation de certains paramètres pour voir s'ils ont une influence sur les performances. Les résultats obtenus ont été comparé avec la marche aléatoire simple (les résultats peuvent s'améliorer si nous avons plus de temps et un bon matériel).

Enfin, dans toutes les simulations, DAL montre ses supériorités sur Random. Cela prouve quantitativement que la stratégie d'exploration proposée est efficace.

Conclusion générale et perspectives

Le foraging Multi-Robots constitue un benchmark dans la robotique mobile. C'est l'abstraction de plusieurs applications du monde réel tel que : le transport, le déminage, le nettoyage, la recherche et le sauvetage...etc. la recherche et le sauvetage consiste à naviguer dans un espace catastrophique selon une stratégie de déploiement dans le but de chercher et transporter des victimes à certaines localisations. Pour concevoir une stratégie de recherche et de sauvetage efficace en temps et en nombre de victimes sauvés, l'exploration de l'environnement ainsi que le sauvetage doivent être aussi rapide.

Nous avons proposé dans ce travail une solution basée intelligence en essaim qui assure une recherche est un sauvetage de plusieurs victimes dispersés aléatoirement dans un environnement. Notre contribution se compose d': (1) une exploration basée Dragonfly et Lévy walk qui permet une large dispersion des robots et qui augmente la chance de détecter plus de victimes ; (2) un sauvetage (autrement dit le transport des victimes au dépôt le plus proche) en utilisant des spots lumineux qui s'étale sur la totalité de l'environnent.

L'algorithme a été implémenté sous la plateforme de robotique mobile ARGoS. Plusieurs simulations ont été réalisé sous la même plateforme et une analyse des résultats montre l'efficacité de notre proposition par rapport à la marche aléatoire.

Comme perspectives à ce travail, nous cherchons à :

1. Améliorer l'algorithme proposé pour prendre en charge la limitation de l'énergie des robots ;
2. Comparer l'algorithme avec des travaux relies de la littérature.

Bibliographie

Abedi, M., & Gharehchopogh, F. S. (2020). An improved opposition based learning firefly algorithm with dragonfly algorithm for solving continuous optimization problems. *Intelligent Data Analysis, 24*(2), 309--338.

Abuomar, L., & Al-Aubidy, K. (2018). Cooperative Search and Rescue with Swarm of Robots Using Binary Dragonfly Algoritlnn. *2018 15th International Multi-Conference on Systems, Signals & Devices (SSD)* (pp. 653--659). IEEE.

Ali, A., Othman, R., Yacob, Y., & Alkanaani, J. (2019). ParametersTuning of Adaptive Firefly Algorithm based Strategy for t-way Testing.

Bahel, V., Peshkar, A., & Singh, S. (2020). Swarm Intelligence-Based Systems: A Review. *Proceeding of International Conference on Computational Science and Applications* (pp. 149--156). Singapore: Springer.

Beni, G., & Wang, J. (1993). Swarm Intelligence in Cellular Robotic Systems. *Robots and biological systems: towards a new bionics?* (pp. 703--712). Berlin, Heidelberg.: Springer.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems* (Vol. 1). Oxford university press.

Campo, A., & Dorigo, M. (2020). *On the Design of Self-Organized Decision Making in Robot Swarms.*

Charrier, R. (2009). *L'intelligence en essaim sous l'angle des systèmes complexes : étude d'un système multi-agent réactif à base d'itérations logistiques couplées Autre [cs.OH].* Doctoral dissertation, Université Nancy 2.

Chen, J., Gauci, M., Li, W., Kolling, A., Groß, R., & others. (2015). Occlusion-based cooperative transport with a swarm of miniature mobile robots. *IEEE Transactions on Robotics, 31*(2), 307-321. Récupéré sur http://playerstage.sourceforge.net/index.php?src=stage

Cruz, D. P., Maia, R. D., & De Castro, L. N. (2019). A critical discussion into the core of swarm intelligence algorithms. *Evolutionary Intelligence, 12*(2), 189--200.

Devi, N., Uma, V., Praveena, T., & Jyothi, M. (2013). Emerging and Challenging Applications In Swarm Robotics. *Int J Eng Sci*, 318--321.

Din, A., Jabeen, M., Zia, K., Khalid, A., & Saini, D. (2018). Behavior-based swarm robotic search and rescue using fuzzy controller. *Computers & Electrical Engineering, 70*, 53-65.

Dorigo, M., & Gambardella, L. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation, 1*(1), 53-66.

Dorigo, M., & Stützle, T. (2019). Ant colony optimization: overview and recent advances. Dans *Handbook of metaheuristics* (pp. 311--351). Springer, cham.

Ducatelle, F., Di Caro, G., Förster, A., Bonani, M., Dorigo, M., Magnenat, S., . . . Others. (2014). Cooperative navigation in robotic swarms. *Swarm Intelligence, 8*(1), 1-33.

Falconi, R. (2009). *Coordinated control of robotic swarms in unknown environments.* Doctoral dissertation, alma.

Garnier, S. (2008). *Decisions collectives dans des systemes d'intelligence en essaim.* Doctoral dissertation, Universite de Toulouse, Universite Toulouse III-Paul Sabatier.

Garnier, S., Gautrais, J., & Theraulaz, G. (2007). The biological principles of swarm intelligence. *Swarm intelligence, 1*(1), 3--31.

Greenhagen, C., Krentz, T., Wigal, J., & Khorbotly, S. (2016). A real-life robotic application of the particle swarm optimization algorithm. Dans C. Greenhagen, *2016 Swarm/Human Blended Intelligence Workshop (SHBI)* (pp. 1-5). IEEE.

Huang, X., Arvin, F., West, C., Watson, S., & Lennox, B. (2019). Exploration in Extreme Environments with Swarm Robotic System. Dans *2019 IEEE International Conference on Mechatronics (ICM)* (Vol. 1, pp. 193--198). IEEE.

Jevtić, A., & Andina de la Fuente, D. (2007). Swarm intelligence and its applications in swarm robotics. *WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics ; 6 CIMMACS ; 6.* Tenerife, España.

Kaur, S., & Sharma, S. (2017). Performance Evaluation of Artificial Bee Colony and Compressive Sensing Based Energy Efficient Protocol for WSNs. *2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, 91-96.

Kornienko, S., & Kornienko, O. (2011). New principles of coordination in large-scale micro-and molecular-robotic groups. *CoRR, abs/1109.3765*.

Kumar , A., Manikutty, G., Bhavani, R., & Couceiro, M. (2017). Search and rescue operations using robotic darwinian particle swarm optimization. Dans *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 1839-1843). IEEE.

Larfi, O., & Aisset, A. (2016). *Developpement d'un algorithme intelligent pour la commande des robots mobiles.*

Li, X., & Clerc, M. (2019). Swarm intelligence. Dans M. Gendreau, & J. Potvin (Éds.), *Handbook of Metaheuristics* (Vol. 272, pp. 353-384). Springer, cham.

Liekna, A., Grundspenkis, J., & others. (2014). Towards practical application of swarm robotics: overview of swarm tasks. *Engineering for rural development, 13*, 271--277.

Mahendra, P., & Pandey, R. (2016). *Swarm Intelligence. In International Conference on Advanced Computing.* Teerthanker Mahaveer University, College of Computing Sciences and Information Technology (CCSIT), Moradabad.

Mayeur, B. (2014). *Outil automatique de placement de robots pour des expériences de Swarm Robotics.* Mémoire présenté en vue de l'obtention Du diplôme du Master en Sciences Année académique 2013 – 2014, UNIVERSITE LIBRE DE BRUXELLES, UNIVERSITE D'EUROPE, Faculté des Sciences Département d'Informatique.

Mosavi, M., Khishe, M., & Ghamgosar, A. (2016). Classification of sonar data set using neural network trained by Gray Wolf Optimization. *Neural Network World, 26*(4), 393.

Nadipally, M. (2019). Optimization of Methods for Image-Texture Segmentation Using Ant Colony Optimization. Dans *Intelligent Data Analysis for Biomedical Applications* (pp. 21-47). Elsevier.

O'Keeffe, J., Tarapore, D., Millard, A., & Timmis, J. (2017). Towards fault diagnosis in robot swarms: An online behaviour characterisation approach. Dans *Annual Conference Towards Autonomous Robotic Systems* (pp. 393-407). Springer.

Palappan, A., & Thangavelu, J. (2018). A New Meta Heuristic Dragonfly Optimizaion Algorithm for Optimal Reactive Power Dispatch Problem. *Gazi University Journal of Science, 4*, 1107-1121.

Patnaik, S., Yang, X.-S., & Nakamatsu, K. (2017). *Nature-Inspired Computing and Optimization* (éd. 1, Vol. 10). heidelberg: Springer.

Pinciroli, C., Trianni, V., O'Grady, R., Pini , G., Brutschy, A., Brambilla, M., . . . others. (2012). ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm intelligence, 6*(4), 271-295.

Sahin, E. (2004). Swarm robotics: From sources of inspiration to domains of application. Dans E. Sahin, & W. Spears (Éds.), *International workshop on swarm robotics* (Vol. 3342, pp. 10-20). Berlin, Heidelberg: Springer.

Srivatsava, P., Mallikarjun, B., & Yang, X.-S. (2013). Optimal test sequence generation using firefly algorithm. *Swarm and Evolutionary Computation, 8*, 44-53.

Suresh, V., Sreejith, S., Sudabattula, S., & Kamboj, V. (2019). Demand response-integrated economic dispatch incorporating renewable energy sources using ameliorated dragonfly algorithm. *Electrical Engineering, 101*(2), 421- 442.

Tan, Y. (2013). Swarm robotics: collective behavior inspired by nature. *J Comput Sci Syst Biol, 6*(6), e106.

Tan, Y., & Zheng, Z.-y. (2013). Research advance in swarm robotics. *Defence Technology, 9*(1), 18-39.

Tang, J., Zhu, K., Guo, H., Gong, C., Liao, C., & Zhang, S. (2018). Using auction-based task allocation scheme for simulation optimization of search and rescue in disaster relief. *Simulation Modelling Practice and Theory, 82*, 132-146.

Yang, X.-S. (2014). Cuckoo search and firefly algorithm: overview and analysis. Dans X.-S. Yang (Éd.), *Cuckoo search and firefly algorithm* (Vol. 516, pp. 1-26). Springer, cham.

Yogeswaran, M., & Ponnambalam, S. (2010). Swarm robotics: An extensive research review.

Zemzami, M., Elhami, A., Makhloufi, A., Elhami, N., Itmi, M., & Hmina, N. (2017). Applying a New Parallelized Version of PSO Algorithm for Electrical Power Transmission. *IOP Conference Series: Materials Science and Engineering, 205*, 12-32.

Zoghby, N., Loscri, V., Natalizio, E., & Cherfaoui, V. (2014). Chapter 8: Robot Cooperation and Swarm Intelligence. Dans *Wireless Sensor and Robot Networks: From Topology Control to Communication Aspects* (pp. 163-201). World Scientific.

Webographie

[1] Cabo Pulmo, URL : http://www.expeditioncruising.com/2018/12/from-aboard-national-geographic-venture_18.html , derniere consultation : 05/09/2020.

[2] А им летать охота (обновлено!), URL : http://29palms.ru/index.php?link=blog&action=showblog&blog=2415 , derniere consultation : 05/09/2020.

[3] Stratford and District Beekeepers Association, URL : https://www.stratfordbeekeepers.org.uk/2016/06/27/our-first-swarm/ , derniere consultation : 05/09/2020.

[4] A Swarm of small red ants, URL : https://www.dreamstime.com/swarm-red-ants-swarm-small-red-ants-go-back-to-their-house-image160467939 , derniere consultation : 05/09/2020.

[5] Echo, URL : https://edusaint.com/study/courses/class-9th-ncert-science-course/lessons/sound-free-lessons-for-class-9th-physics/topic/sonar-reflection-loudness-pitch-echo-explained/ , derniere consultation : 05/09/2020.

[6] A self-organizing thousand robot swarm, URL : https://www.seas.harvard.edu/news/2014/08/self-organizing-thousand-robot-swarm , derniere consultation : 05/09/2020.

[7] Chris Joel, URL : https://www.3dprintersonlinestore.com/could-an-army-of-3d-printed-drones-destroy-america , derniere consultation : 05/09/2020.

[8] Mbiyimoh Ghogomu, URL : http://thehigherlearning.com/2015/11/11/why-millennials-fear-the-future-part-iii-the-wild-world-of-human-enhancement/ , derniere consultation : 05/09/2020.

[9] Rougeole Epidémiologie, URL : http://rougeole-epidemiologie.overblog.com/2014/10/les-antivax-ne-savent-pas-sur-quel-pied-danser.html , derniere consultation : 05/09/2020.

[10] Stage/2D multiple-robot simulator...", URL : http://playerstage.sourceforge.net/stage/stage.html , derniere consultation le 02 / 03 / 2020 à 10 h 05.

[11] Gerkey B. et al.," Stage/2D multiple-robot simulator...",
URL : http://playerstage.sourceforge.net/index.php?src=stage , derniere consultation le 02 / 03 / 2020 à 10 h 05.

[12] Cyberbotics Webots, URL : https://www.youtube.com/watch?v=uEV4iq11-jQ , derniere consultation le 02 / 03 / 2020 à 10 h 11.

[13] Cyberbotics ltd,"Webots User Guide ...", URL : https://cyberbotics.com/doc/guide/introduction-to-webots , derniere consultation le 02 / 03 / 2020 à 10 h 11.

[14] Gazebo, URL : https://medium.com/teamarimac/integrating-sonar-and-ir-sensor-plugin-to-robot-model-in-gazebo-with-ros-656fd9452607 , derniere consultation le 02 / 03 / 2020 à 10 h 20.

[15] OSRF,"Gazebo Tutorials / Beginner: Overview ...", URL
: http://gazebosim.org/tutorials?tut=guided_b1&cat=#WhatisGazebo? , derniere consultation le 02 / 03 / 2020 à 10 h 20.

[16] Gazebo, URL : http://playerstage.sourceforge.net/gazebo/gazebo.html , derniere consultation le 26 / 04 / 2020 à 19 h.