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 :

Une exploration par essaim de robots pour la recherche de cibles dans des environnements inconnus

---

Encadré Par :

Dr. Ouarda ZEDADRA

Présenté par :

Aymen BENZAID

Septembre 2020

Dédicace

Je tiens en tout premier lieu à remercier ***Allah****.*

Je voudrais dédier ce travail :

A ma très chère mère wassila

La source de tendresse et l'exemple du dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Ta prière et ta bénédiction m'ont été d'un grand secours pour mener à bien mes études. Aucune dédicace ne saurait être assez éloquente pour exprimer ce que tu mérites pour tous les sacrifices que tu n'as cessé de me donner depuis ma naissance, durant mon enfance et même à l'âge adulte. Tu as fait plus qu'une mère puisse faire pour que ses enfants suivent le bon chemin dans leur vie et leurs études. Je te dédie ce travail en témoignage de mon profond amour. Puisse Dieu, le tout puissant, te préserver et t'accorder santé, longue vie et bonheur.

A mes amies :

Betihi Haythem et islam e rami et danawachi et hazem et Laadassi Iman et mido.

A mon Encadreur *ZEDADRA Ouarda.*

A mes camarades et Mes collègues

Ziou Abderrezzag et Salah Eddine Merzougui et Nor El Houda Maizi et Qui m'ont aidé dans la réalisation de ce travail

Et à tous ceux que j'aime et à toutes les personnes

Qui m'ont prodigué des encouragements et se sont donné la peine de me soutenir durant cette formation.

Enfin je le dédie aux *enseignants et ma promo et mes collègues d'informatiques.*

Benzaid Aymen

Remerciement

A terme de ce mémoire que nous achevons grace à Dieu tout puissant.

Mes sincères remerciements et ma reconnaissance vont à tous ceux qui m'ont aidé tout au long de mon parcours

A Dr. ZEDADRA Ouarda, pour son assistance, sesprécieux conseils et sa disponibilité constante ainsi que le savoir qu'elle m'a transmis.Je voudrai lui adresser mes vifs remerciements et de lui témoigner mon sincère reconnaissance.

Je tiens ensuite

A remercier mes parents pour le soutien inconditionnel dont ils ont fait preuve depuis que mon projet est défini. Merci pour le soutien financier, moral, psychologique et matériel. Merci à tous les amies et les membres de la famille pour leur aides leur soutien et leurs encouragements dans les moments difficiles.

Je remercierai

*Mes amis et camarades de la promotion **2020** pour ces cinq années passées ensemble, dans les meilleurs moments comme dans les pires.*

Je remercie également l'honorable jury pour l'intérêt qu'il porte à juger notre travail

Résumé

L'exploration Multi-Robots consiste à naviguer dans l'espace selon une stratégie de déploiement dans le but de chercher des objets précieux. Dans cette tâche la dispersion des robots sur des zones distinctes peut donner de résultats efficaces : la recherche sera plus rapide, le nombre d'objets collectés sera plus élevé et les collisions seront évitées. Les méta-heuristiques inspirées des techniques collectives des insectes sociaux ont montré une grande flexibilité et efficacité dans la coordination des essaims de robots. Plusieurs algorithmes inspirés de l'intelligence en essaim assurent un comportement de synchronisation soit par attraction des robots vers des zones d'intérêts (comme les phéromones des fourmis, la dance des abeilles ou même la lumière des lucioles) ou par répulsion des robots de certaines zones critiques.

Dans ce travail, nous cherchons à assurer une grande dispersion de robots dans leurs environnements en utilisant la logique contraire de la lumière des lucioles. Au lieu de suivre la lumière la plus élevée, les robots doivent s'éloigner de la lumière pour ne pas rentrer dans une zone d'exploration d'un autre robot et donc assurer une grande couverture. L'objectif de l'exploration dans notre travail est de localiser le maximum possible des objets (destructive foraging). À cet effet, nous avons utilisé l'algorithme de lucioles (fireflies- FA) avec une modification de son comportement d'attraction par un comportement de répulsion. Ces algorithmes ont été utilisé en hybridation avec une marche aléatoire simple (Random Bounce) et une marche stratégique (Global Lawnmower). 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 nos propositions.

Mots clés : intelligence en essaim, robotique en essaim, exploration Multi-Robots, Algorithme Firefly (FA), Random Bounce (RB), Global Lawnmower (GL)

Sommaire

Remerciement ii
Résumé iii
Sommaire iv
Liste des figures vi
Liste des tableaux vii
Liste des équations viii
Introduction générale 1
Chapitre 1 : Intelligence et robotique en essaim 3
I Introduction 3
II Intelligence en essaim 3
II.1 Introduction à l'intelligence en essaim 3
II.2 Interactions et transferts d'information dans les systèmes d'intelligence en essaim 5
II.3 Fonctions assurées par les systèmes d'intelligence en essaim 6
II.4 Applications de l'intelligence en essaim 7
II.5 Principaux algorithmes issus de l'intelligence des essaims 8
III Robotique en essaim 9
III.1 Définitions 9
III.2 Caractéristiques 11
III.3 Les avantage de la robotique en essaim 11
III.4 Comportement de base d'un essaim de robots 11
III.5 Applications de la robotique en essaim dans le monde réel 13
III.6 Plateformes de simulation de robotique en essaim 14
IV Conclusion 15
Chapitre 2 : Exploration Multi-Robots – Synthèse des travaux 16
I Introduction 16
II Exploration Multi-Robots 16
II.1 Définitions 16
III Synthèse des travaux 17
IV Conclusion 34
Chapitre 3 : Conception 35

I. Introduction 35
II. Problématique, objectifs et proposition 35
III. Les Algorithmes reliés 36
III.1. Algorithme Firefly (FA) 36
III.2. Algorithme Random Bounce (RB) 38
III.3. Algorithme global lawn-mower (GL) 38
III.4. Algorithme Spiral carré 39
IV. Algorithmes proposés 40
IV.1 Description des algorithmes RBF et GLF 40
IV.2 Pseudos codes des algorithmes RBF et GLF 42
IV.3 Machine d'état des robots 43
V. Conclusion 44
Chapitre 4 : Implémentation et Simulations 46
I Introduction 46
II Environnement de développement 46
III Modélisation des composants de notre système 48
III.1 Caractéristiques du Foot-bot utilisé dans nos algorithmes 48
III.2 Analyse du contrôleur des algorithmes RBF et GLF 49
IV Simulations et analyse des résultats 49
IV.1 Critères de performance 50
IV.2 Scénarios de simulation 50
V.3 Résultats, discussions et comparaisons 50
V Conclusion 57
Conclusion générale et perspectives 58
Bibliographie 59
Webographie 64

Liste des figures

Chapitre 1 :

Figure 1.1 : Exemples de systèmes qualifiés d'intelligents en essaim. 5
Figure 1.2 : Un exemple d'exécution de l' ACO 8
Figure 1.3 : Principe de fonctionnement du PSO. 8
Figure 1.4 : Les types d'abeille dans un ABC et leur fonctionnement 9
Figure 1.5: Exemples de systèmes qualifies d'essaim de robots 10
Figure 1.6 : Agrégation. 12
Figure 1.7 : Exploration 12
Figure 1.8 : Déplacement cordonnés. 12
Figure 1.9 : Transport collectif. 12
Figure 1.10 : Foraging. 13
Figure 1.11 : Recherche et sauvetage. 13
Figure 1.12 : l'interface de argOs 14
Figure 1.13 : le simulateur Player/Stage 14
Figure 1.14 : le simulateur Webots 15
Figure 1. 15 : le simulateur Gazebo 15

Chapitre 2 :

Figure 2.1 : Exploration multi-robots. 17

Chapitre 3 :

Figure 3.1 : comportement d'attraction des lucioles 37
Figure 3.2 : La trajectoire prise par un seul robot sous l'algorithme de recherche par rebond aléatoire (RB) 38
Figure 3.3 : global lawn-mower 39
Figure 3.4: Spiral carré. 39
Figure 3.5 : Machine d'état des algorithmes RBF et GLF 43

Chapitre 4 :

Figure 4.1 : L'architecture du simulateur argos. 47
Figure 4.2 : Influence de nombre de robots sur les performances. 51
Figure 4.3 : Une exécution sous ARGoS de l'algorithme RBF avec 50 robots. 52
Figure 4.4 : Influence de la taille de l'environnement sur les performances. 53
Figure 4.5 : Une exécution sous ARGoS de l'algorithme GLF avec taille de l'environnement 140 m². 53
Figure 4.6 : Influence du nombre de clusters sur les performances. 54
Figure 4.7 : Une exécution sous ARGoS de l'algorithme RBF avec 8 cluster (A) et 2 cluster (B) 55
Figure 4.8 : Influence de la distribution des cibles sur les performances. 56
Figure 4.9 : Une exécution sous ARGoS illustrant la distribution des cibles : cluster (A), aléatoire (B) 56

Liste des tableaux

Chapitre 2 :

Tableau 2.1 : Comparaison qualitative des travaux reliés. 33

Chapitre 4 :

Tableau 4.1: Scenarios de simulations réalisés..... 50
Tableau 4.2: Influence du nombre de robots sur les performances. 51
Tableau 4.3: Influence de la taille de l'environnement sur les performances. 52
Tableau 4.4: Influence du nombre de clusters sur les performances. 54
Tableau 4.5: Influence de la distribution des cibles sur les performances. 56

Liste des équations

Chapitre 3 :

Equation (3.1) **:** Variation de l'intensité lumineuse selon la distance r 37
Equation (3.2) **:** Equation de distance entre deux lucioles 37
Equation (3.3) **:** Fonction d'attractivité 37
Equation (3.4) **:** Equation de mouvement d'une luciol 37
Equation (3.5) : Fonction de génération de nouvelle direction 38
Equation (3.6) **:** Fonction de génération de longeur l d'un robot GLF 40
Equation (3.7) **:** Fonction d'augmentation de l'intensité lumineuse 40
Equation (3.8) **:** Fonction de diffusion de l'intensité lumineuse 40
Equation (3.9) : Fonction de mouvement d'un robot RBF\GLF evitier lumiere 41
Equation (3.10) : Vitesse d'un robot RBF\GLF dans l'états evitment d'obstacle 41

Introduction générale

Ce travail se situe dans le cadre général de l'étude des systèmes d'intelligence en essaim. Ce champ de recherche s'attache à comprendre et à concevoir des mécanismes décentralisés de résolution collective de problèmes. Ces mécanismes sont inspirés de certains comportements collectifs spectaculaires comme la construction de nids chez les termites et les fourmis, la coordination du mouvement dans les vols d'oiseaux et les bancs de poissons. Les individus ont des capacités de calcul, de mémorisation et de décisions minimales mais montrent des comportements collectifs fascinantes. Ces mécanismes permettent aux sociétés animales d'organiser de manière efficace l'activité de grands groupes d'individus.

Ces systèmes ont servi de source d'inspiration pour la conception de plusieurs systèmes artificiels comme le routage de marchandise, les animations graphiques. Ils ont aussi contribué au développement d'une nouvelle branche de la robotique qui s'en inspire pour coordonner l'activité de plusieurs robots afin qu'ils accomplissent collectivement une tâche, reconnu sous le nom *robotique en essaim*.

L'exploration des environnements inconnus et l'une des problématiques fondamentaux de la robotique mobile. Elle constitue la phase préliminaire pour plusieurs applications de robotique comme le nettoyage, la recherche et le sauvetage, le foraging…etc. Dans ce travail, nous utilisons l'exploration pour la localisation du maximum possible des objets (destructive foraging).

Le présent mémoire propose une contribution pour résoudre efficacement le problème d'exploration (pour une recherche de targets) en utilisant un essaim de robots coordonnés à l'aide d'un algorithme issu de l'intelligence en essaim (Firefly- FA). Dans sa version originale, le FA utilise l'intensité de lumière élevée pour attirer les individus les uns aux autres. Dans notre travail, nous utilisons la lumière plutôt pour repousser les robots pour assurer une grande dispersion dans l'environnement. L'algorithme a été utilisé avec une marche aléatoire simple (random bounce) et une marche stratégique (global lawnmower). L'algorithme a été implémenté sous la plateforme de robotique mobile ARGoS et les simulations réalisées montrent la valeur de la contribution.

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 de travaux reliés au problématique d'exploration avec un tableau de comparaison de travaux ;
- Le troisième chapitre présente la problématique, les objectifs et la proposition. Puis explique les algorithmes reliés. Ensuite, décrit en détail les deux algorithmes améliorés avec la machine d'état des robots ;
- Le quatrième chapitre présente l'implémentation, les simulations réalisées, les résultats obtenues et leurs discussions.

Nous terminerons ce manuscrit par une conclusion et quelques perspectives.

Chapitre 1 : Intelligence et robotique en essaim

I Introduction

L'Intelligence en essaim, désigne l'apparition de phénomènes cohérents à l'échelle d'une population dont les individus agissent selon des règles simples. L'interaction entre actions individuelles simples peut de façons variées permettre l'émergence de formes, organisations, ou comportements collectifs, complexes ou cohérents, tandis que les individus eux se comportent à leur échelle indépendamment de toute règle globale. Une des branches les plus actives de l'intelligence en essaim est la robotique en essaim. C'est l'étude de la façon de faire collaborer des robots par l'application des méthodes de l'intelligence en essaim pour résoudre une tâche, qui serait autrement impossible à résoudre par un seul robot tel que : le nettoyage, le déminage, l'exploration des environnements hostiles, la recherche et le sauvetage.

Ce chapitre se trouve divisé en deux parties. Dans la première, nous focalisons sur l'intelligence en essaim où nous présentons quelques définitions, les propriétés d'un système qualifié d'intelligent en essaim, la communication dans un système d'intelligence en essaim, les domaines d'application et quelques algorithmes issus de l'intelligence en essaim. Tandis que, dans la deuxième partie, nous présenterons la robotique en essaim, à savoir quelques définitions, les principales caractéristiques d'un essaim de robots, les avantages de l'utilisation des robots en essaim, leurs applications aux monde réel, et nous terminerons par quelques plateformes de simulations de robotique mobile.

II Intelligence en essaim

Dans cette première partie, nous focalisons sur l'intelligence en essaim. Nous faisons un survol sur quelques définitions, les concepts reliés, les caractéristiques et les domaines d'application.

II.1 Introduction à l'intelligence en essaim

1) Naissance et définitions

Le mot "*essaim*" évoque l'image d'un grand nombre de petits insectes où chaque individu effectue une tâche simple, mais dont l'action produit un comportement complexe dans son ensemble **[Kennedy et al., 2001].** Les essaims sont définis comme des collections de nombreux individus simples qui

interagissent avec les autres individus et l'environnement en utilisant un contrôle décentralisé et auto-organisée pour atteindre leurs objectifs **[Miller Peter, 2007].**

Définition 1.1 : Charrier et al. **[Charrier et al., 2007**], définissent l'intelligence en essaim en informatique comme : *« l'intelligence artificielle en essaim recouvre un ensemble d'algorithmes utilisés en informatique qui ont la particularité de faire interagir de multiples processus élémentaires de façon décentralisée, c'est-à-dire sans contrôleur global, afin de résoudre des problèmes ».*

Définition 1.2 *:* Merkle et Blum **[Blum &Merkle, 2008]** proposent une définition pragmatique de l'intelligence en essaim du point de vue informatique **«** *L'intelligence en essaim est une discipline de l'intelligence artificielle moderne qui traite de la conception de systèmes Multi-Agents en vue d'applications telles que l'optimisation et la robotique ».*

La **Figure 1.1** montre quelques exemples de systèmes qualifiés d'intelligents en essaim, observés dans des sociétés d'animales est des insectes sociaux.

		
Essaim de Libellule [1]	Essaim de chauves-souris [2]	Essaim de Pingouins [3]
		
Essaim de Loups gris [4]	Essaim d'oiseaux [5]	Essaim de criquets [6]

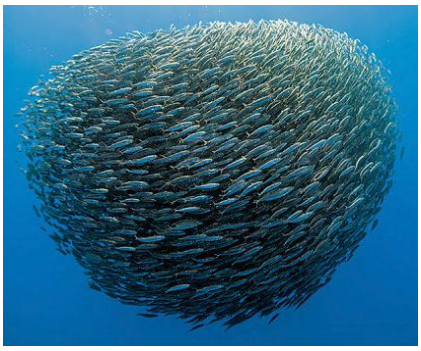
		
Banc de poisson [7]	Essaim de fourmis [8]	Essaim d'abeilles [9]

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

2) Propriétés des systèmes d'intelligence en essaim

Il y a plusieurs propriétés qui caractérisent les systèmes d'intelligence en essaime. En particulier, on distingue classiquement les propriétés suivantes **[Garnier, 2008]** :

- **La dynamique :** Les interactions permanentes entre les individus signifient l'évolution du système au fil du temps. Le système peut atteindre un état stable mais il ne sera jamais gelé. Cette fonctionnalité indique que le système est flexible et peut s'adapter à tout changement dans l'environnement.
- **La redondance :** Certaines tâches peuvent être effectuées par des nombreuses individus différents. Par conséquent, le système augmente ses chances de voir une tâche importante mise en œuvre même si l'individu ne l'accomplit pas ou disparaît tout simplement. Cette caractéristique permet au système d'être robuste.
- **L'émergence:** Le comportement de groupe est le résultat d'un ensemble non linéaire de contributions de chaque membre. Par conséquent, le comportement qui peut résulter d'un système d'essaim ne se limite pas aux capacités individuelles de ses membres.

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

Il existe des interactions directes et indirectes (ou stigmergiques).

- **Transfert indirect d'information :** La transmission indirecte d'informations dépend de l'individu qui modifie l'environnement dans lequel il vit, de sorte que deux individus interagissent

indirectement lorsque l'une modifie l'environnement et que l'autre réagit en fonction du nouvel environnement. Cette réaction est un exemple de la stigmergie. Ce terme, qui signifie travail, est une 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. Chaque fois qu'un individu exécute une action, cela entraîne un ajustement de l'espace local. La nouvelle norme entraînera d'autres comportements ultérieurs et les comportements d'autres individus de la colonie. Ce processus conduit à une coordination presque complète du travail d'équipe et peut donner l'impression que la colonie suit un plan précis **[Zoghby et al., 2014]**.

- **Transfert direct d'information :** Les individus collectent leurs informations en remarquant le comportement de leurs voisins. L'information circule sous plusieurs formes : visuelle, tactile ou auditive, mais elle ne dure pas dans l'environnement. De nombreux comportements d'auto-organisation dépendent des interactions directes entre les membres du groupe. Par exemple, les bancs de poissons et les volées d'oiseaux, dans lesquels des milliers voire des millions d'individus se déplacent de manière cohérente, changent de direction soudainement mais simultanément. Cette cohésion quasi parfaite est le résultat d'une forte simulation comportementale couplée à un transfert direct d'informations : chaque oiseau de ces groupes dépend de la direction et de la vitesse de déplacement de ses voisins. Progressivement, chacun s'aligne sur ses voisins qui s'alignent sur ses voisins : ainsi le groupe adopte une direction de mouvement commune sans consulter ses membres ni en superviser un. **[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. Ces emplois peuvent être classés en trois catégories **[Garnier, 2008]** :

1) **Coordination :** c'est l'organisation nécessaire pour résoudre un problème en effectuant les tâches requises au bon endroit et au bon moment. Elle affecte la chronologie et / ou la distribution spatiale des activités individuelles. En d'autres termes, on peut dire que l'exécution d'une tâche à un endroit et à un moment précis et la propagation des comportements dans la population conduisent nécessairement à l'organisation spatiale et temporelle des activités des individus.

2) **Collaboration :** correspond à la répartition appropriée des activités entre individus ou groupes d'individus, chacun selon sa spécialité. C'est également la division du travail où les différences morphologiques pour les individus peuvent contrôler la répartition des tâches entre les individus : Les individus ayant des capacités physiques différentes effectueront différentes tâches. L'expérience joue également un rôle dans la spécialisation des individus. Les individus qui rencontrent régulièrement une certaine incitation dans l'exercice de leurs fonctions, ce qui les rend plus sensibles à l'existence de ce stimulus. Seuls ces individus "spécialisés" peuvent accomplir la tâche associée même à une petite quantité de cette stimulation. Le mécanisme basé sur l'expérience des individus est à l'origine de l'organisation du travail chez plusieurs types d'insectes sociaux.

3) **Délibération :** Les délibérations correspondent aux nombreuses possibilités dont dispose le système, car il choisit une alternative parmi ces possibilités. C'est également le choix de groupe ou de décision de groupe. Par exemple, une abeille peut choisir la parcelle de fleurs la plus productive en recrutant des travailleurs inactifs. Les fourmis peuvent également choisir parmi la plupart des sources de nourriture les plus riches et parmi les nombreuses voies d'accès à la source, qui est la plus courte.

II.4 Applications de l'intelligence en essaim

Le domaine de l'intelligence en essaim est très riche, il existe de nombreuses applications, concernant les drones, les robots (micro et nano), optimisation, le domaine médical et la sécurité, data mining, les problèmes de prévision... Ci-dessous, nous fournissons deux de ces exemples :

1) **Résolution de problèmes d'optimisation :** La difficulté réside sur la question du choix et de la régulation d'une méta-heuristique pour un problème donné qu'il ne semble pas y avoir de consensus sur ce point, dans tous les cas l'objectif reste d'obtenir la meilleure solution possible à moindre coût de calcul [**Charrier,2009]**.

2) **Robotique d'essaim :** La robotique en essaim est actuellement l'une des principales applications de l'intelligence en essaim. La robotique en essaim ajoute les contraintes du monde physique à celle de la théorisation des phénomènes en essaim. Sous un autre angle de vue, elle propose un terrain d'expérimentation et de validation stimulant pour les algorithmes de l'intelligence en essaim. **[Charrier, 2009].**

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

Ci-dessous, nous présentons les trois algorithmes issus de l'intelligence en essaim les plus populaire. Il y on a beaucoup d'autres algorithmes dans la littérature.

1) L'optimisation des Colonies de Fourmis (Ant Colony Optimization- ACO)

ACO est une méta-heuristique, où une colonie artificielle coopère pour trouver de bonnes solutions pour séparer les problèmes **[Charrier, 2009].** La collaboration est le composant principal des algorithmes ACO. En fait, ces algorithmes allouent des ressources de calcul à un certain nombre de fourmis qui communiquent indirectement par la stigmergie (voir **Figure 1.2**). Les algorithmes ACO peuvent être utilisés pour résoudre des problèmes d'optimisation statique et dynamique.

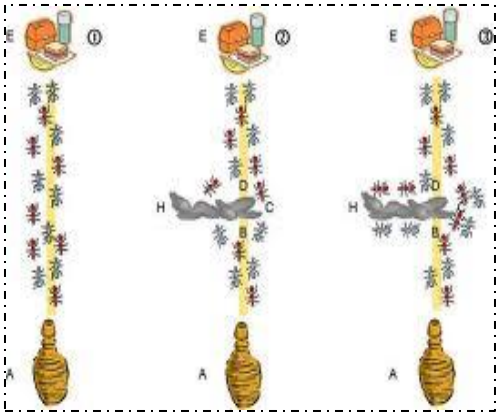


Figure 1.2 : Un exemple d'exécution de l' ACO **[10]**

2) Optimisation par Essaims Particulaires (Particle swarm optimization- PSO)

Il a été suggéré par Kennedy et Eberhart **[Kennedy et Eberhart., 1995]**. C'est un algorithme d'optimisation basé sur la population. Son mécanisme fonctionne en initialisant au hasard un ensemble de solutions possibles, puis l'optimisation se fait à plusieurs reprises. Dans l'algorithme PSO, la position optimale peut être trouvée en suivant les meilleures particules (voir **Figure 1.3**).

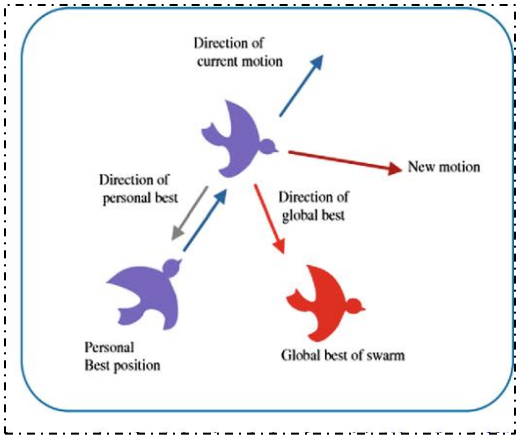


Figure II1.3 : Principe de fonctionnement du PSO **[Esmin et al.,2015]**

Cet algorithme est très approprié pour des problèmes réels tels que : calcul évolutif, optimisation, télécommunications, le contrôle, l'extraction de données, le traitement du signal et bien d'autres. Bien que PSO ait été principalement utilisé pour résoudre les problèmes à objectif unique sans restrictions, les algorithmes PSO ont été développés pour résoudre les contraintes, les problèmes d'optimisation multitâche, les problèmes de changement de scène dynamique.

3) Colonies d'Abeilles Artificielles (Artificial Bee Colony- ABC)

En 2005, Karaboga a proposé Artificial Bee Colony(ABC) **[Karaboga., 2005].** L'algorithme est inspiré du comportement de collecte des abeilles et réaliser des recherches ciblées en divisant la colonie d'abeilles (voir **Figure 1.4**). Il a des propriétés de paramétrage simples et faciles à implémenter. L'algorithme peut résoudre une série de problèmes d'optimisation, notamment l'optimisation des fonctions, l'optimisation des problèmes de compatibilité, etc.

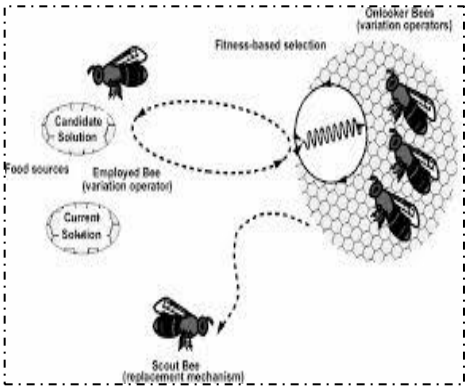


Figure 1.4 : Les types d'abeille dans un ABC et leur fonctionnement **[Sharma et al., 2012]**

III Robotique en essaim

La robotique en essaim est l'une des applications les plus active de l'intelligence en essaim. Comme son nom l'indique, il met en œuvre un ensemble de robots de capacités limitée qui, grâce au système de contrôle décentralisé, résolvent des problèmes complexes.

III.1 Définitions

Définition 1.3: Beni **[Beni, 2004],** définit l'intelligence dans un système de robotique en essaim comme: « *une aptitude à générer des schémas ordonnés de façon imprévisible, schémas sur lesquels une analyse est faite par chaque robot pour optimiser si besoin ses actions en vue d'atteindre un objectif collectif fixé à l'avance*. »

Définition 1.4: Dorigo et Sahin **[Dorigo & Sahin, 2004],** définit la robotique en essaim comme : « *l'étude de la façon dont un grand nombre d'agents relativement simples incarnes physiquement peuvent être conçus pour qu'un comportement voulu puisse émerger des interactions locales entre agents et entre les agents et l'environnement* »

Nous présentons dans la **Figure 1.5**, un ensemble de systèmes qualifiés de robots en essaim :

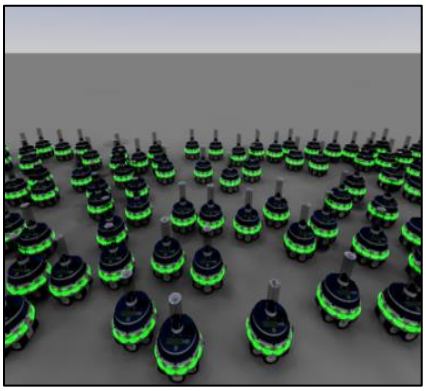
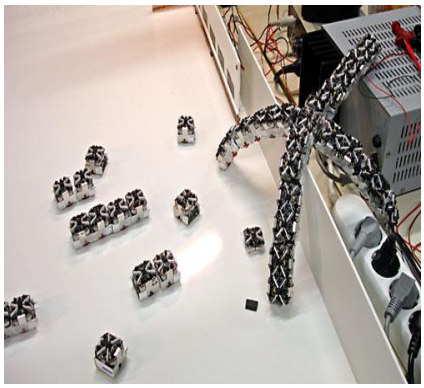
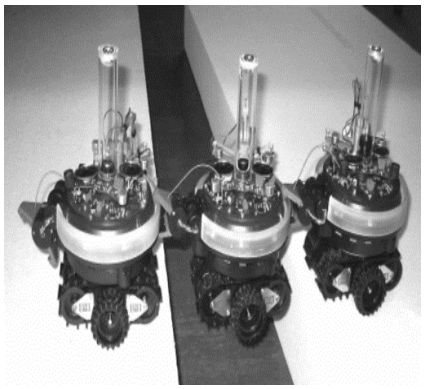
		
Foot-bot **[11]**	EPuck **[12]**	Drones **[13]**
		
Nano-bots **[14]**	Swarm bots **[15]**	Micro-robots **[16]**
		
Swarm cyborg **[17]**	Swarm bots **[18]**	Swarm cyborg **[19]**

Figure 1.5: Exemples de systèmes qualifies d'essaim de robots

III.2 Caractéristiques

Les principales caractéristiques des essaims de robots **[Sahin et al., 2008]** sont résumées dans : la robustesse, la scalabilité et flexibilité.

- **Robustesse :** le système a un haut degré de tolérance aux pannes. Pratiquement, si un robot échoue dans l'exécution de sa tâche, le système évoluera dans une nouvelle configuration qui rétablira le bon fonctionnement du système
- **La flexibilité :** nécessite que le système ait la capacité de générer des solutions modulaires pour différents problèmes.
- **Scalabilité :** l'augmentation du nombre de robots ne dégrade pas les performances du système.

III.3 Les avantage de la robotique en essaim

Selon Charrier **[Charrier, 2009],** Les avantages d'un système d'essaim de robot sont :

- Alternative bon marché : les appareils sont simples, faciles à construire et moins chers qu'un seul robot puissant.
- Des unités simples sont plus faciles à produire en masse et sont interchangeables, voire jetables.
- Le système en essaim est plus faible, plus robuste aux diverses pannes ou perturbations auxquelles il peut être soumis du fait de la redondance des unités.
- Il est doté d'importantes capacités d'adaptation à son environnement.
- Il peut résoudre des problèmes complexes du fait de ses multiples unités de calcul, bien au-delà des problèmes que peuvent résoudre les entités individuellement.
- La communication entre les robots est réduite en raison des interactions indirectes qui offrent la possibilité d'améliorer les performances des unités.
- Une faible complexité des unités et une réduction des couts par rapport aux systèmes robotique traditionnels.
- La recherche dans ce domaine montre qu'un ensemble de comportements individuels relativement primitifs amélioré par la communication produira un large ensemble de comportements d'essaims complexes.

III.4 Comportement de base d'un essaim de robots

Théoriquement, les robots en essaim ont été utilisés pour étudier, résoudre et optimiser certains comportements. Nous présentons dans cette section, les plus pertinents de ces comportements :

1) **Agrégation :** permet de regrouper tous les robots d'un essaim dans une région de l'environnement (voir ***Figure 1.6***). L'agrégation est très utile dans les taches de construction, car elle permet aux robots dans un essaim d'être proches afin qu'ils puissent interagir les uns avec les autres [**Soysal et Şahin., 2006].**

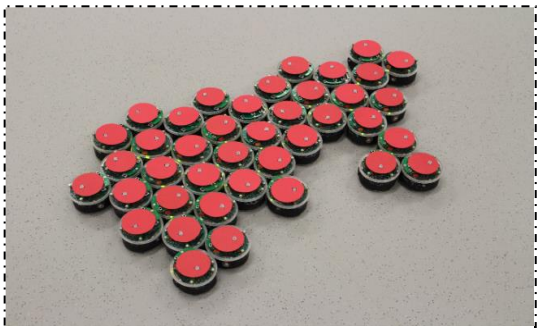


Figure 1.6 : Agrégation **[li et al.,2016].**

2) **Exploration :** L'exploration de l'environnement est l'une des tâches les plus étudiées dans le champ de la robotique en essaim. Cette tache peut être divisée en trois parties : (1) dispersion, (2) couverture et (3) localisation de cibles (voir **Figure 1.7**). L'objectif d'un processus d'exploration est de couvrir tout l'environnement dans un temps raisonnable.

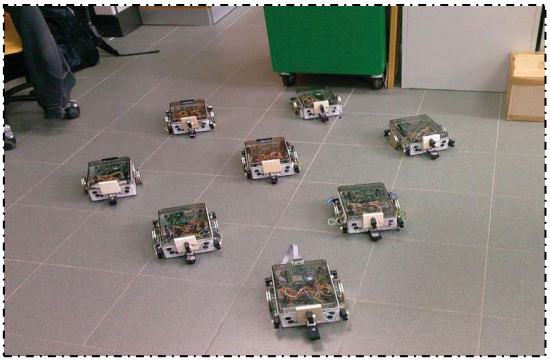


Figure 1.7 : Exploration **[Gasparri et al., 2012]**

3) **Déplacement coordonnée :** Les robots adoptent ici une direction de mouvement commune au sein d'un groupe. Les robots doivent être capables d'estimer à plusieurs reprises la direction du mouvement des autres membres du groupe afin qu'elle reste constante, c'est-à-dire la connexion réelle entre les robots les uns avec les autres (voir **Figure 1.8**) **[Kaminka et al, 08].**

Figure 1.8 : Déplacement cordonnés **[Garnier, 2008].**

4) **Transport coopératif :** Manipulation coopérative quand au moins deux robots sont nécessaires pour traiter un objet (voir **Figure 1.9**). Un simple mouvement d'un objet peut être trop important pour être déplacé par un seul robot. Mais la manipulation peut également être plus compliquée [**Ijspeert et al., 2001].**

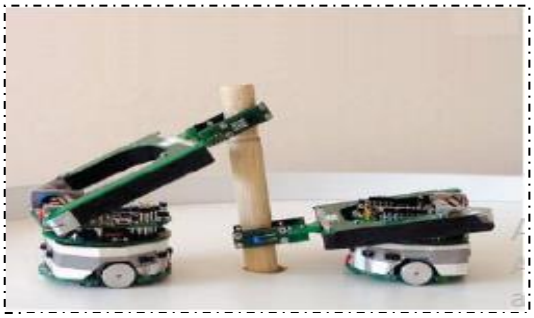


Figure 1.9 : Transport collectif **[Martinoli, 1999].**

5) **Foraging :** Une tâche de foraging est analogue à la recherche d'une cible, elle ajoute seulement la nécessité de ramener les cibles à un dépôt. Les robots recherchent des objets, les ramassent et les amènent dans une certaine zone cible, le foraging peut être considérée comme un comportement de recherche et de récupération (voir **Figure 1.10) [Hamann, 2018].**

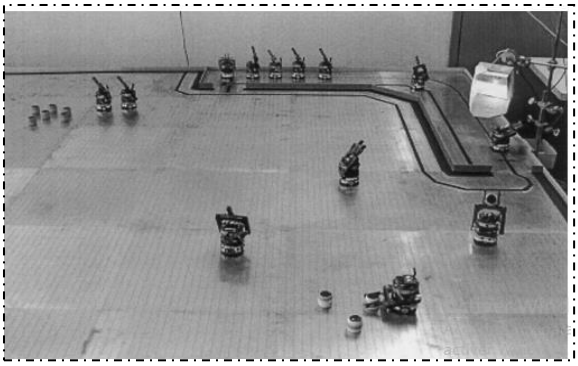


Figure 1.10 : Foraging **[Krieger & Billeter, 2000].**

6) **Rechercher et sauvetage :** Cartographie des zones après des catastrophes naturelles, des accidents, des explosions ..., afin de rechercher des survivants et localiser les sources de dangers (voir **Figure 1.11**) tels que la pollution toxique, les fruites dans les canalisations, la radioactivité... etc. **[Jevtić et Andina,20 07] [Sahin, 04].**

Figure 1.11 : Recherche et sauvetage.

III.5 Applications de la robotique en essaim dans le monde réel

De ces comportements de base, plusieurs applications du monde réel peuvent être réaliser **[Bahel et al., 2020]** :

- **Militaire :** détection, désarmement élimination des bombes qui sont des taches dangereuses pour les humains. Une sorte d'essaim de robots à grande vitesse qui peut pénétrer profondément à l'intérieur du corps de l'intrus aux frontières militaires afin de détruire les vaisseaux sanguins menant éventuellement à la mort.
- **Sciences spatiales :** La NASA utilisait auparavant la robotique d'essaim pour alimenter les missions d'exploration de l'espace.
- **Médecine :** Utilisation de nano-robots basé sur la technologie de la robotique en essaim qui se déplaçant dans les veines et les artères humaine pour combattre certains types de tumeurs.
- **Agriculture :** La robotique en essaim, en particulier, est considérée extrêmement pertinente pour les applications d'agriculture à grande échelle. Plus précisément, dans les scenarios de cartographie décentralisée, et la détection des mauvaises herbes dans un champ par un groupe de petits robots.

L'essaim de robots a été utilisé dans de nombreuses autres applications : surveillance de l'environnement d'auto-examen pour étudier les normes environnementales, identification des sources de dangers tels que les fuites de produits chimiques ou de gaz, la pollution toxique, les fuites de pipelines et de rayonnements, exécution de tâches de surveillance et élimination des déchets toxiques et exploration et cartographie.

III.6 Plateformes de simulation de robotique en essaim

1) **Autonomous Robots Go Swarming ARGoS:** C'est un simulateur Multi-Robots open source (voir **Figure 1.12**). Son objectif est la simulation de grands essaims hétérogènes de robots. ARGoS, suit une approche modulaire approfondie qui permet à la fois à l'utilisateur d'ajouter facilement des fonctionnalités personnalisées et d'allouer des ressources mathématiques lorsque l'expérience est nécessaire. L'une des caractéristiques uniques d'ARGoS est la capacité d'exécuter plusieurs moteurs physiques de différents types et de cartographier différentes parties de l'environnement. Les robots peuvent passer d'un moteur à un autre de manière transparente **[Pinciroli et al.,2012]**.

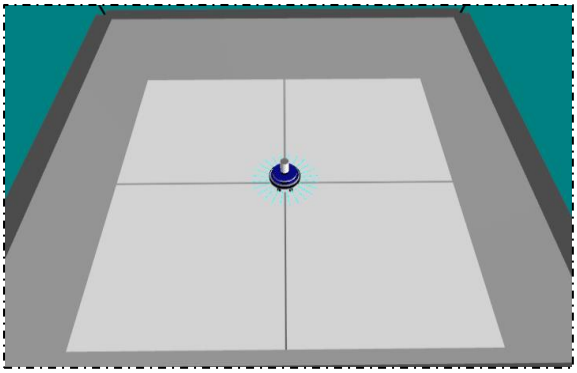


Figure 1.12 : L'interface de ARGOS **[20]**

2) **Player / Stage :** Player/Stage est un projet open source permettant la simulation du comportement d'un ou plusieurs robots dans un environnement simulé ainsi que la commande d'un ou plusieurs robots réels sur l'ordinateur. Player n'est pas un simulateur mais une couche d'abstraction entre un robot physique et un programme. Stage est un simulateur 2D sur lequel Player peut s'appuyer (voir **Figure 1.13**). Stage peut accepter un ou plusieurs robots permettant ainsi de simuler les comportements collaboratifs **[21]**.

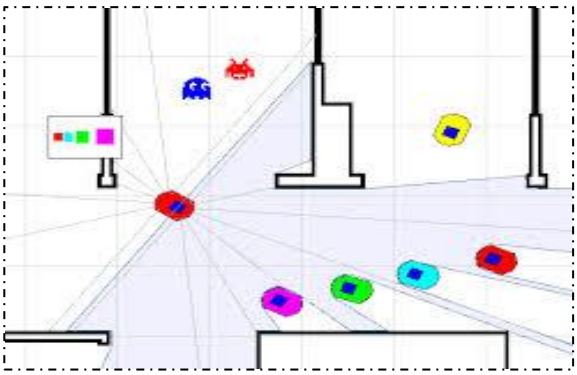


Figure 1.13 : Le simulateur Player/Stage **[21]**

3) **Webots :** Une application de bureau open source et multiplateforme utilisée pour simuler des robots (**Figure 1.14**). Il fournit un environnement de développement complet pour modéliser, programmer et simuler des robots. Il a été conçu pour un usage professionnel et est largement utilisé dans l'industrie, l'éducation et la recherche. Cyberbotics Ltd. Maintient Webots comme son principal produit en continu depuis 1998 **[22]**.

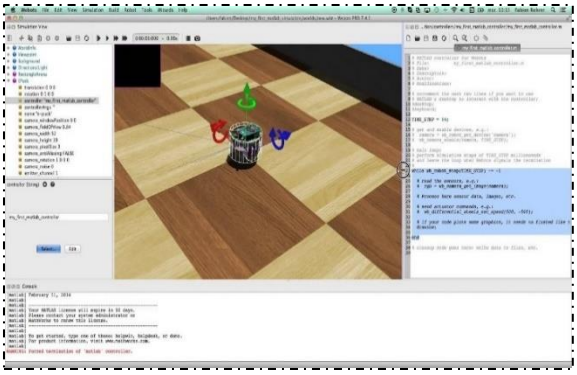


Figure 1.14 : Le simulateur Webots **[22]**

4) **Gazebo :** Gazebo est un simulateur Multi-Robot open source. Comme stage, il est capable de simuler un ensemble de robots, capteurs et objets, mais le fait dans un monde tridimensionnel (**Figure 1.15**). Il génère des informations réalistes pour les capteurs et simule les interactions physiques entre les objets (en intégrant un moteur physique) **[23]**.

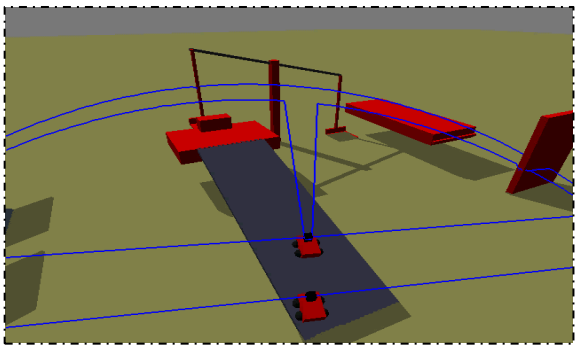


Figure 1. 15 : Le simulateur Gazebo **[23]**

IV Conclusion

Ce chapitre a été divisé en deux parties. Dans la première, nous avons présenté quelques définitions sur le domaine de l'intelligence en essaim. Les propriétés d'un système qualifié d'intelligent en essaim, la communication dans un tel système, les domaines d'application et quelques algorithmes issus de l'intelligence en essaim. Dans la deuxième partie, nous avons présenté quelques définitions sur le domaine de la robotique en essaim ainsi que les principales caractéristiques d'un essaim de robots, les avantages des robots en essaim, les applications au monde réel, puis nous avons fourni une liste des plateformes de simulation de robotique mobile.

Cette première partie été très bénéfique pour la compréhension des deux domaines intelligence et robotique en essaim ainsi que leur caractéristiques et propriétés. Elle nous a aidé à fixer l'application a traité (exploration Multi-Robots) ainsi que la plateforme de simulation adéquate (ARGoS). Dans le chapitre suivant, nous étudions le problème d'exploration Multi-Robots et nous réalisons une synthèse de travaux qui va nous aider à proposer une technique d'exploration efficace.

Chapitre 2 : Exploration Multi-Robots – Synthèse des travaux

I Introduction

L'exploration a comme but la couverture d'un endroit inconnu, ou de l'identifier, de le découvrir ou de l'enregistrer. Il s'agit d'une étape préliminaire et fondamentale dans diverses applications tels que : l'exploration spatiale, la recherche et le sauvetage, le nettoyage, le foraging...etc. Les méthodes et moyens d'exploration diffèrent en fonction de la différence d'exploration, soit par visualisation, soit par recherche et étude, et comprend l'utilisation d'un ensemble d'outils, de matériaux et d'appareils tels que des robots, des caméras, des enregistreurs, des microscopes, des cartes et des boussoles, en plus d'autres matériaux et outils.

Nous commençons ce chapitre par quelques définitions de l'exploration Multi-Robots, puis on passe à une synthèse des travaux reliés qui se termine par un tableau comparant les différents travaux.

II Exploration Multi-Robots

L'exploration d'un environnement inconnu est un problème fondamental dans les robots mobiles qui permet de créer une carte utile dans de nombreuses applications (sécurité, assistance, etc.). Une bonne stratégie d'exploration crée une carte complète ou partielle à un temps rapide. Dans ce qui suit, nous présentons quelques définitions de l'exploration Multi-Robots. La **Figure 2.1** montre un ensemble de robots qui explorent un environnement avec des obstacles.

II.1 Définitions

Définitions 2.1 : *'Le processus d'exploration Multi-Robots consiste à couvrir efficacement toutes les zones inconnues de l'environnement. Cette tâche soulève de nombreux défis car elle fonctionne en équipe pour effectuer des tâches difficiles dans des environnements inconnus, extrêmement dangereux pour les humains et ces robots peuvent accomplir la tâche spécifique plus rapidement, moins cher et plus efficace* ***[Kaldé et al., 2014]****'.*

Définitions 2.2 : *'Explorer des environnements inconnus est l'une des principales applications des systèmes Multi-Robots. Il s'agit d'explorer des zones inconnues et de détecter certaines cibles aléatoirement réparties dans ces zones dans le minimum de temps possible. Par conséquent, il est essentiel que les robots soient déployés dans différentes régions afin de pouvoir couvrir efficacement toutes les zones inconnues de l'environnement qui permet de construire une carte utile dans de nombreuses applications dans différents domaines, de l'industriel, comme la tonte de gazon ou l'aspirateur, au militaire comme le déminage, ou encore humanitaire, comme les opérations de recherche et de sauvetage* ***[Benavides et al.,2016]****'.*

L'exploration Multi-Robots basé intelligence en essaim est basée sur le comportement biologique et la communication indirecte. Elle s'inspire généralement du comportement de certains types d'animaux, comme les fourmis, qui utilisent des substances chimiques appelées phéromones pour induire des changements de comportement chez d'autres membres de la même espèce **[Kaldé et al., 2014]**.

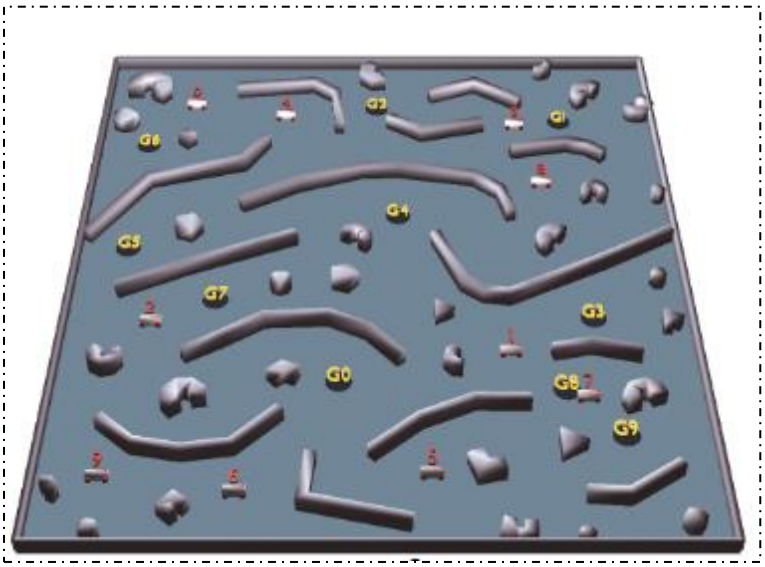


Figure 2.1 : Exploration Multi-Robots **(Le et plaku., 2018).**

III Synthèse des travaux

L'exploration Multi-Robots des environnements inconnus a été considéré largement dans la littérature. Nous présentons dans cette section les travaux récents et pertinents de ce problème. Nous avons étudié et résumé 17 articles, puis nous avons comparé ces derniers à travers un ensemble de critères utile pour montrer les différences et les similarités entre ces travaux.

1) A Swarm Robotic Exploration Strategy Based on an Improved Random Walk Method [Pang et al., 2019]

Pang et al. **[Pang et al., 2019]** proposent une amélioration de la stratégie d'exploration aléatoire, dans laquelle chaque robot ajuste la taille de son pas régulièrement pour réduire le nombre de zones déjà explorées en se basant sur une estimation de la densité des robots dans une zone particulière. L'intervalle de temps entre deux cas d'inférence physique est utilisé pour estimer la densité de robots ; plus l'intervalle de temps est court, plus la densité est élevée. Lorsque la densité est relativement élevée, chaque robot explore une région relativement petite avec une taille de pas relativement court ; lorsque la densité est faible le robot explore une région relativement grande avec une taille de pas large.

En ajustant leurs tailles de pas de mémoire adaptative et en contrôlant leurs directions d'exploration, les robots se répartissent de façon uniforme dans l'environnement et chaque robot explore sa propre zone, améliorant ainsi l'efficacité d'exploration.

Les auteures ont réalisé deux type de tests : (1) des simulations informatiques avec le simulateur webots, (2) des expérimentations avec de vrai robots. Dans les premières simulations, ils ont utilisé 1, 10, 20 et 30 robots pour une période de 20 minutes. Au démarrage les robots sont placés au centre de l'environnement et déplacent avec une vitesse de 0.1 m/s. Le critère de performance utilisé est le ratio de couverture. Chaque simulation est répétée 20 fois et le résultat moyen avec l'écart type ont été marquées. La méthode proposée donne de meilleure résultats par rapport à la marche Brownian et à la marche Lévy Flight.

Dans les expérimentations réelles, les auteurs utilisent de vrai robots (e-puck) mais en raison de difficulté, le critère de performance utilisé a été remplacé par le nombre d'objets trouvés. Les comparaisons avec BW et LF donner des résultats supérieurs.

2) A Darwinian Swarm Robotics Strategy Applied to Underwater Exploration [Sanchez et al., 2018]

Sanchez et al. **[Sanchez et al., 2018]** ont proposé une stratégie Multi-Robots autonome pour explorer des environnements sous-marins inconnus dont les dimensions maximales sont connues a priori en collectant des données sur les propriétés de l'eau et l'existence d'obstacles. La stratégie développée est basée sur la méthode Robotic Darwinian Particle Swarm Optimization (RDPSO), qui a été initialement conçue pour les applications robotiques terrestres. Il s'agit de la première étude à généraliser l'algorithme RPSO pour les applications 3D, avec concentration sur la robotique sous-marine.

RDPSO fait une hybridation de DPSO (Darwinian Particle Swarm Optimization) et RPSO (Robotic Particle Swarm Optimization). L'introduction de concepts évolutifs améliore l'algorithme en le rendant moins susceptible d'être piégé au niveau des optima locaux. Le système est divisé en plusieurs sous-essaims qui recherchent collectivement les optima globaux. Chacun des sous-essaims agit individuellement suivant le RPSO, mais ils sont également soumis à un ensemble de règles qui imitent la sélection naturelle. En plus des multiples essaims existants, le système comprend un groupe de robots exclus socialement, dont les membres ne suivent pas l'algorithme RPSO, mais adoptent plutôt une marche aléatoire. Les membres de l'essaim communiquent les informations qu'ils collectent à un robot parent unique, qui à son tour stocke les données reçues, mis à jour la carte d'exploration, puis partage la carte avec tous les membres. Le robot parent ne participe pas à l'exploration et il se charge seulement de fournir aux membres les informations dont ils ont besoin pour travailler efficacement.

Le système est représenté comme une grille d'occupation formée de cellules cubiques. Chacune des cellules a une valeur numérique qui lui est attribuée (0 au début de l'exploration), indiquant si elle a été explorée ou non ; 0 indique que la cellule n'a pas encore été explorée, tandis que 1 indique qu'elle a déjà été explorée par au moins un membre de l'essaim. La direction de chaque membre de l'essaim est calculée sur la base de l'unité vectorielle qui indique la position actuelle du robot vers la cellule inexplorée la plus proche de la carte. De cette façon, le robot se déplacera dans une direction qui améliore la fonction de performance, qui dans cette application représente l'exploration du scénario. La cellule inexplorée la plus proche est calculée par chaque robot sur la base de la carte reçue du robot parent et de la position actuelle du robot, qui est supposé connue.

La stratégie d'essaim proposée a été testée dans un environnement de simulation créé avec Gazebo. Les résultats montrent que le temps d'exploration diminue avec de grande taille d'essaim.

3) Autonomous Exploration Guided by Optimization Metaheuristic [Santos et al., 2018].

Les auteurs proposent un stratège d'exploration basée sur une fonction d'utilité continue Multimodale et une optimisation méta-heuristique. Le problème de l'exploration est formulé comme un problème d'optimisation, qui produit des données pour un algorithme basé Firefly qui est apte à rechercher des points dans l'espace de la solution, représentant des cellules sur la carte en cours de construction.

L'environnement a exploré est modélisé comme une fonction objective à 2-dimensions, dans laquelle les points d'énergie minimale sont potentiellement les meilleures cibles pour le robot. Une méta-heuristique

basée sur l'algorithme Firefly est également proposée pour minimiser la fonction objective, et trouver la meilleure cible à atteindre par le robot à chaque mouvement.

La tâche d'exploration peut être décomposée dans les directives suivantes :

- Rechercher les frontières connues et inconnues (maximiser le gain d'information) ;
- Réaliser des itinéraires en utilisant les informations formées par les caractéristiques de la carte (minimiser le risque de collision) ;
- Eloigner des zones précédemment explorées de la carte (minimiser l'effort répétitif).

Les directives précédentes sont suivies au moyen d'un comportement qui pondère chaque directive. La fonction objective est définie ainsi comme la somme pondérée des fonctions la composants (des trois directives) pour la sélection de la position suivante, une minimisation de la fonction objective est demandée, dans la mesure de ne pas prendre en considération tous les cellules de la carte pour réduire le temps de calcul. Cela est fait en utilisant l'algorithme Firefly.

La proposition a été implémentée sous la plateforme robotique (V-REP) sous le système ROS (Robot Operating System). Deux environnements de test ont été utilisé chacun de taille 10 m. Trois formulations ont été utilisés comme critères : CLF (Conditional loss), LLF (linear loss), PLF (propositionnel loss) pour évaluer la distance entre le point d'intérêt P et la position courante du robot.
L'exploration guidée par le PLF donne de bonne résultats en temps et en longueur de la trajectoire.

4) Cooperative Search and Rescue with Swarm of Robots Using Binary Dragonfly Algorithm [Abuomar et Al-Aubidy.2018].

Abuomar et Al-Aubidy proposent **[Abuomar et Al-Aubidy,2018]**, un algorithme de recherche et de sauvetage par un essaim de robots. L'algorithme fait une amélioration de l'algorithme d'optimisation binary dragonfly (BDA). L'extension RBDA (Robotic Binary Dragonfly Algorithm) concerne l'ajout de deux comportements : évitement d'obstacles, et la communication.

BDA comporte cinq comportements de base :

1. *Séparation :* concerne l'évitement d'obstacles et tout autre individu ;
2. *Alignement :* concerne la correspondance de vitesse des individus ;
3. *Cohésion :* concerne le rapprochement vers un centre d'une zone ;
4. *Attraction :* concerne l'attraction a une zone d'intérêt ;

5. *Distraction :* concerne le comportement de fuir de toute contrainte qui influence négativement le suivi des individus.

Avec la modification apportée dans l'équation de vecteur des pas, l'algorithme RBDA a été applicable dans un environnement avec obstacles et avec des contraintes de communication.

L'algorithme RBDA a été testé sur des fonction de Benchmark (sphere, rosenbrock, De Jong, Grie Wank, Rastrifin, Ackliu) et comparé avec l'algorithme BDA d'origine. Les résultats montrent que RBDA minimise l'erreur pour converger à la solution optimale. Les auteurs ont simulé l'algorithme sur le Toolbox virtuel SIMROBOT sous Matlab. Un environnement simulé de 300 x 300 mètres avec des obstacles aléatoires distribuées a été utilisé avec un nombre de robots qui varie entre 5 et 15 déployés aléatoirement. Un robot victime est déployé aussi aléatoirement. RBDA recherche l'amélioration de la solution optimale qui est la victime. Les tests sur la position des robots pour atteindre la solution optimale sont faites. Ces améliorations été rapide et la convergence vers la solution optimale est faite dans des temps meilleures. Ainsi, l'augmentation dans le nombre de robots diminue le nombre des itérations pour atteindre la solution optimale.

5) Swarm robotic network using Lévy flight in target detection problem [Katada et al., 2016].

Les auteurs traitent le problème de détection de cible dans des environnements fermés. Une fois un robot détecte une cible, les robots communiquent immédiatement avec une station de base via des robots de relais intermédiaires en utilisant une communication sans fil. Ce travail, étudie les performances de deux algorithme d'exploration aléatoire : la marche aléatoire usuelle et le Lévy Flight afin d'observer et conclure l'effet de la taille du pas de la marche aléatoire et du nombre de robots pour un problème de détection de cible.

Les auteurs proposent une architecture de subsumption a trois couche : Transmission, évitement d'obstacles et exploration de cible.

- Dans la **couche exploration**, le module d'exploration envoie un message à l'un des trois modules : vers l'avant, tournez à droite, tournez à gauche.
- Dans **la couche évitement d'obstacle**, le module détection d'obstacle envoie un message soit au module tournez à droite, soit tournez à gauche selon les entrées sensorielles des capteurs à distance ;

- Dans **la couche transmission**, le module de cible envoie un message au module transmit messages et au module STOP quand les entrées sensoriales des capteurs infrarouge de détection d'une cible sont superviseure a un seuil ;
- Le module **transmit message** envoie un message à la base via des robots de relais.

Les auteurs ont implémenté réellement dans le couloir du bâtiment de Université Setsunan à Niagawa trois prototypes : Random walk 2 (RN2) dans laquelle la taille du pas est 1, Random walk (6) dans laquelle la taille du pas est 6. Lévy flight (LF6) utilise deux phases séparées le mouvement en avant et la rotation. La taille de pas dans le mouvement en avant est déterminée selon une distribution de probabilité Lévy. La taille du pas moyen est effectivement 6 (la taille maximum du pas est initialisé a 30 fixée expérimentalement). La cible définie comme une boule émettrice infrarouge est placée dans le coin inferieur gauche. Dans le coin supérieur droit, une station sans fil a été placée, cette station c'est un ordinateur portable équipé du même X Bee que celui des robots. Elle est définie comme un coordinateur. Les robots sont toujours placés à la même position initiale qui est le coin inferieur, droit à côté de la station dans des directions aléatoires. La cible est placée à une distance de 80 mètres de la station et les robots ont besoin d'être toujours connectés à la station via le réseau de l'essaim de robot.

Dans les expérimentations, les auteurs changent le nombre de robot de 10, 15 à 20, et une expérimentation se termine soit quand la station reçoit un message soit après 1800 secondes sans réception de message, la stratégie d'exploration a été changée à chaque fois de RN (2), RN (6) et LF (6), et 10 expérimentations ont été réalisé pour chaque configuration. Le taux de succès est très important et c'est le meilleure dans la LF (6).

6) Collective-Adaptive Lévy Flight for Underwater Multi-Robot Exploration [Sutantyo et al., 2013].

Les auteurs proposent une nouvelle stratégie d'exploration pour localiser efficacement des cibles dans des environnements sous-marins. Cette stratégie se base sur les algorithmes Firefly et Lévy Flight (LF). La stratégie proposée adapte les mouvements des robots en fonction de la distribution des cibles dans l'environnement. Les auteurs ont proposé une adaptation du LF pour générer des longueurs de nage ou lieu des longueurs de pas. Elle inclut une interaction sociale entre les robots sans avoir une communication globale ni stigmergique. Les robots sont dotés d'une mémoire à court terme qui représente leur expérience individuelle. Cette expérience est définie comme l'attractivité et elle est utilisée comme la communication locale entre robots dans le voisinage. L'algorithme proposé est basé

sur l'algorithme FA original. La première modification concerne le paramètre d'attraction qui est augmenté à chaque fois une cible est localisée et qui diminue avec le temps. Ce paramètre d'attraction influence la vitesse de mouvement des robots ; la deuxième modification concerne l'utilisation de l'algorithme LF pour le mouvement, l'élément aléatoire dans l'équation du FA est remplacé par le nombre aléatoire issue du générateur de distribution Lévy.

L'algorithme commence par initialiser la population avec des positions aléatoires et des vitesses constantes, puis il passe à l'évaluation : qui calcule le paramètre d'attraction, si une cible est détectée augmenter l'attraction sinon diminue l'attraction, si un robot dans le secteur de vision avec une attraction plus élevée, suivi le et modifier l'attraction et la vitesse.

Les simulations ont été réalisé sous COCORO'S, un simulateur sous-main 3D implémenté sous NETLOGO pour tester les performances quand on change le nombre de robots et quand on change la distribution des cibles (cluster, distribuée) l'algorithme a été comparé avec LF, marche aléatoire basée Gaussien et distribution aléatoire. La moyenne de 100 répétition a été envisagée pour les résultats l'algorithme proposé donne les bons résultats. Aussi, les auteurs montrent une expérimentation réelle avec LiLy cocoro dans un Aquarium de 2,5 x 15 mètres, et toujours l'amélioration de LF donne de bonne résultats avec le temps moyen minimal d'exploration.

7) Multi-Robot Cooperative Tasks using Combined Nature-Inspired Techniques [Palmieri et al., 2015].

Les auteures traitent le problème de recherche et désarmement de plusieurs mines distribuées dans un environnement par un ensemble de robots. Ils proposent un algorithme qui hybride les algorithmes ACO et FA (FTS-RR), et l'utilisation d'une autre méthode basée sur l'ACO nommée ATS-RR.

Dans cette stratégie les robots commencent à se distribuer dans les cellules afin que chaque robot explore toutes les cellules indépendamment en utilisant la stratégie ACO, il dépose une phéromone sur chaque cellule qu'il visite, lorsqu'il se déplace à les cellule voisine, la phéromone diminue jusqu'à sa disparation. Les robots choisissent la cellule qui contient la plus petite quantité de phéromones.

Lorsqu'un robot trouve une arme, un certain nombre de robots sont nécessaires pour la désarmer :

- Le robot agit comme une luciole pour attirer un groupe de robots. À chaque pas de temps, la luciole est plus susceptible d'être choisie lorsque sa valeur de l'intensité est élevée. La valeur de l'intensité dépend de la distance. Le robot désigné se dirige donc vers la luciole / robot le plus brillante. En utilisant les informations relatives à l'emplacement des cibles trouvées, le robot extrait la distance

entre celle-ci et les coordinateurs, puis utilise cette distance pour choisir la meilleure cible, qui est généralement plus proche.

- Une fois qu'un robot détecte une mine par lui-même ou reçoit des demandes des autres, il doit prendre la décision de rechercher dans une nouvelle zone ou d'aller vers l'emplacement de mine pour coopérer avec les autres. Dans ce cas, le robot qui détecte une mine devient un coordinateur et commence à attirer le nombre nécessaire de robots à l'emplacement des mines. Les robots coordinateurs déposent une phéromone, différente de celle utilisée pour l'exploration ; ce type de phéromone attirent d'autres robots pour les guider à la cellule qui contient les mines.

Les robots qui perçoivent la phéromone essaient d'atteindre la position de la mine en préférant la cellule avec une concentration plus élevée de phéromone, ce qui signifie que les cellules sont probablement plus proches des emplacements de la mine. Les robots, en dehors de la zone des phéromones, poursuivent d'explorer la région.

Grâce aux simulations, les auteurs constatent que la FTS-RR pourrait mieux fonctionner en terme de temps pour terminer la mission et le nombre de visites dans la cellule, résultant en une meilleure distribution des robots avec un meilleur temps combiné pour désarmer.

8) Bio-inspired Strategies for the Coordination of a Swarm of Robots in an Unknown Area [Palmieri et al., 2015].

Les auteurs traitent le problème de la recherche et le désarmement des mines réparties dans un environnement par un groupe de robots. Ils ont proposé un algorithme hybride utilisant l'ACO et le FA appelé FTS-RR, ainsi que l'ATS-RR (voir le résumé précédent **[Palmieri et al. 2015]**.

Dans ce travail, les auteurs concentrent sur l'application de ces méthodes dans des environnements simulés et en les comparant à l'algorithme PSO. Les auteurs ont analysé la qualité de la solution en utilisant une métrique de performance spécifique : erreur relative indiquant le nombre d'accès dans les cellules qui donnent une mesure de l'efficacité avec laquelle les robots sont répartis dans la zone. Les simulations ont montré qu'en appliquant l'ACO+FA l'erreur relative est faible.

9) Discrete Firefly Algorithm for Recruiting Task in a Swarm of Robots [Palmieri et al., 2016].

Les auteurs proposent une hybridation entre les algorithmes FTS-RR et ATS-RR, pour résoudre le problème de la recherche et désarmement des mines avec un groupe de robots. La méthode a été expliquée dans le résumé précédent, où les auteurs ont appliqué ces méthodes et les ont comparées au

PSO **[Palmieri et al., 2015]**. Par comparaison, il a été constaté que le FTS-RR donne de meilleurs résultats que PSO et ATS-RR. Ils ont donc effectué plusieurs expérimentations avec la FTS-RR en Java dont l'environnement est en grille de taille égale.

Plusieurs paramètres doivent être modifiés dans chaque simulation, à savoir : le nombre de cibles, le nombre de robots, la taille de l'environnement. Les résultats des tests montrent que les valeurs des paramètres sont importantes car la complexité des tâches augmente en fonction de la taille de l'environnement et du nombre de mines. En particulier, il est préférable dans ce cas d'équilibrer l'attractivité des lucioles avec des mouvements aléatoires afin de mieux répartir les robots dans la zone et éviter toute répétition dans toute zone nécessitant une augmentation du temps de réalisation toutes les tâches.

10) Comparison of bio-inspired algorithms applied to the coordination of mobile robots considering the energy consumption [Palmieri et al., 2017].

Dans **[Palmieri et al., 2017]** les auteurs comparent des hybridations de l'algorithme ACO avec FA, avec PSO, puis avec ABC pour localiser et collaborer à désarmer des mines qui sont distribuées dans l'espace. Les trois proposition (ACO+FA, ACO+PSO, ACO+ABC) ont été testé à travers un ensemble d'expériences utilisant un simulateur basé sur Java. Lrs robots ont un stock d'énergie limité. Pour l'hybridation ACO, FA (FTS-RR) cette méthode a été expliquée dans le résumé précédent **[Palmieri et al., 2015]**.

Pour l'hybridation ACO, ABC, lorsque le robot ne reçoit qu'une seule candidature, il se déplace vers celle-là. Si le robot reçoit plus d'une demande, il doit déterminer la cible vers laquelle il doit se déplacer. Dans ce cas, il utilise le concept des ABC. Essentiellement, lorsque le robot k dans la cellule x reçoit un paquet d'un coordinateur dans la cellule x1, le coût de la cible z pour le robot k à l'étape t est calculé comme la distance traditionnelle entre le robot et la cible.

Autrement dit, chaque cible est associée à une probabilité qui est choisisse parmi un ensemble d'objectifs découverts. Une fois toutes les côtés calculés, le robot sélectionnera la cible en tournant la roue. Ensuite, le robot se déplacera. Cette technique de coordination est parfaitement adaptée, pour éviter que de nombreux robots s'approchent de la même cible et disperse les robots sur différents sites.

Avec l'hybridation ACO, PSO, lorsqu'un robot trouve une arme, les robots s'appuient sur un algorithme PSO pour désarmer la cible. Les simulations ont montré que la consommation d'énergie est plus élevée

pour l'approche PSO, surtout lorsque la taille de l'essaim est petite et que la dimension et le nombre de cibles sont élevés en l'environnement.

Les méthodes FA et ABC sont compatibles lorsque la tâche n'est pas complexe, mais la différence est plus évidente lorsque le nombre de cibles augmente et que le nombre de robots dans l'environnement est petit. Par conséquent, le mécanisme de coordination devient plus complexe pour des tâches complexes, et la stratégie basée FA donne généralement de meilleures performances.

11) Self-adaptive decision-making mechanisms to balance the execution of multiple tasks for a Multi-Robots team [Palmieri et al., 2018].

Palmiri et al. Ont développé une stratégie basée sur l'hybridation de l'algorithme ACO et FA, pour résoudre un problème d'optimisation restreint à deux cibles où les robots mobiles doivent effectuer deux tâches spécifiques d'exploration et en même temps collaborer et coordonner pour désarmer des cibles dangereuses.

La stratégie proposée a été testée à travers un ensemble d'expériences utilisant un simulateur basé Java, L'environnement d'exécution est divisé en cellules à espacement égal contenant un certain nombre de mines statiques et dynamiques et un ensemble d'obstacles fixes avec un budget énergétique limité (dans chaque expérience, le nombre de robots, de mines, d'obstacles et d'énergie sont déterminés).

Les robots commencent à se distribuer dans les cellules afin que chaque robot explore toutes les cellules indépendamment en utilisant la stratégie ACO, le robot dépose une phéromone dans la cellule qu'il a découverte, et lorsqu'il se déplace aux cellules voisines, la quantité de phéromone diminue jusqu'à sa disparition avec le temps.

Lorsqu'un robot localise une arme, un certain nombre de robots sont nécessaires pour la désarmer se robot commence à agir comme une luciole pour attirer un groupe de robots. À chaque pas de temps, la luciole est plus susceptible d'être choisi lorsque son intensité de lumière est élevée. La valeur de l'intensité dépend de la distance. Le robot désigné se dirige donc vers le luciole / robot le plus brillant la gravité diminue avec l'augmentation de la distance. S'il existe de nombreuses lucioles (les robots demandent de l'aide), les robots qui répondent à la demande doivent choisir la cible à désarmera. En se déplaçant à la cible, un robot peut réaliser une exploration des zones par les peules il passe qui est généralement la plus proche. Les résultats des expériences ont montré que le choix du robot pour l'une des deux tâches (exploration et coopération pour le désarmement) n'est pas clair, car la comparaison

entre les deux cibles est fortement liée au nombre de cibles déployées dans la zone par rapport à la taille de l'essaim de robots.

12) Swarm robotics in wireless distributed protocol design for coordinating robots involved in cooperative tasks [De Rango et al., 2018].

De Rango et al. **[De Rango et al., 2018]** ont développé une stratégie basée sur l'algorithme ACO nommée ATS-RR, et une autre approche utilisant le modèle WIFI pour communiquer avec les autres. Ils proposent un protocole sans fil distribué inspiré de la nature pour recruter le robot nécessaire sur le site minier afin de limiter le trafic, pour résoudre un problème d'optimisation restreint à deux cibles où les robots mobiles doivent effectuer deux tâches spécifiques d'exploration et en même temps collaborer et coordonner pour désarmer des cibles dangereuses. L'algorithme ATS-RR a été déjà expliqué dans le résumé précédent **[Palmieri et al., 2015]**. Alors que pour communiquer, le robot qui détecte une cible envoie des messages d'annonce transmissent par les autres robots afin que les informations sur la cible peuvent se propager à tout l'essaim.

Les résultats de simulation réduit le temps de convergence par rapport à l'IAS-SS (Inverse Ant System-Based Surveillance System). De plus, l'augmentation du nombre de mines sur le terrain augmente légèrement le temps de convergence moyen tandis que l'augmentation de la zone de recherche (cellules) affecte légèrement les performances du système.

13) Exploration in extreme environments with swarm robotic system [huang et al., 2019].

L'objectif principal de ce travail été la mise en œuvre d'un scénario d'exploration avec un essaim de robots hétérogènes qui peuvent détecter différents types de cibles, simulant différents niveaux de rayonnement. Les auteurs ont étudié l'influence des paramètres expérimentaux sur l'efficacité de l'essaim. La plateforme de simulation utilisée dans ce travail été Mona.

L'algorithme d'exploration, est divisé en trois étapes :

1. *Marche aléatoire :* le robot avance à une vitesse constante et explore en permanence l'environnement à l'aide de son système sensoriel.
2. *Obstacle détecté :* le robot modifie la vitesse des moteurs gauche et droite en fonction de ses capteurs. Par conséquent, il change de direction et poursuit sa marche aléatoire.
3. *Cible détectée :* le robot s'arrête pour une contre période pour investiguer le contenu de la cible détectée. Il enregistre la position relative de la cible et continue la marche aléatoire.

Les auteurs ont utilisé une simulation des rayonnements prévenant, des bidons de pétrole ou d'huile à l'aide de la plateforme V-REP. Il existe différents niveaux de rayonnement et ils doivent être détectés séparément. Pour cela, trois versions du robot Mona ont été conçu où chacun ne peut détecter qu'un type simulé avec couleurs différentes. Les performances ont été testé selon deux critères : le nombre total de fois de détection des cibles, période de simulation nécessaire pour localiser tous les bidons. Les résultats de simulation ont été analysé à l'aide de l'analyse of variance (ANOVA) qui utilise des valeurs clés pour les prouver ou les réfuter.

Les auteurs ont utilisé deux scenarios de simulation : incrémentation de la population (nombre de robots) et changement de la configuration environnementale. Les résultats ont montré que l'augmentation du nombre de robots augmente légèrement le nombre total de cibles détectées. Les résultats obtenus ont été analysée par ANOVA. Elles montrent que l'addition d'un système par 30% moyenne. Alors que, la complexité de l'environnement n'influence pas significativement les performances.

14) Bat Algorithm for Coordinated Exploration in Swarm Robotics [Suárez et Iglesias, 2017]

Les auteurs proposent une nouvelle stratégie d'exploration inspirée de chauve-souris. Les auteurs ont concentré sur les Microbats, qui utilisent un sonar appelé localisateur d'écho, avec des taux d'émission et d'intensité sonore variables, pour détecter les proies, éviter les obstacles et localiser les fissures glaciales dans l'obscurité.

L'algorithme est considéré comme un nombre premier d'individus P (chauves-souris). Toutes les chauves-souris, qui sont une solution potentielle au problème d'optimisation, ont un emplacement x_i et une vitesse v_i. L'algorithme initialise ces variables avec des valeurs aléatoires dans l'espace de recherche. Ensuite, la fréquence d'impulsion et le volume sont calculés pour chaque raquette individuelle. Après cela, l'essaim évolue séparément à travers les générations, comme les états temporels jusqu'à ce que le nombre maximum de générations soit atteint, G_{max}. Pour chaque génération g et chaque palette, la nouvelle fréquence, position et vitesse sont calculées. La meilleure solution actuelle, qui est obtenue en évaluant la fonction objective de toutes les chauves-souris et en classant leurs valeurs de fitness. Les meilleures solutions actuelles et la solution locale qui les entoure sont choisisses de manière prospective selon certains critères. Après cela, la recherche est intensifiée par une marche aléatoire locale. Pour cette recherche locale, une fois la solution choisie parmi les meilleures solutions actuelles, elle est localement perturbée par la marche aléatoire du modèle.

Les résultats expérimentaux étaient bons. Les auteurs conclus que les robots devraient rester coincés dans des configurations difficiles comme les formes U et V, dans lesquelles les unités motorisées sont obligées de revenir, dans les embouteillages dus au surpeuplement, ou dans les voies étroites et les voies où un robot peut avancer à la fois.

15) Random Walks in Swarm Robotics: An Experiment with Kilobots [Dimidov et al., 2016)].

Dimidov et al, **[Dimidov et al., 2016]** ont analysé l'efficacité des modèles de marche aléatoire pour un essaim de Kilobots cherchant une cible statique dans deux conditions environnementales différentes impliquant un espace délimité ou ouvert. Ils étudient l'efficacité de la recherche et la capacité de diffuser des informations au sein de l'essaim par le biais de simulations numériques et d'expériences réelles avec des robots.

Les auteurs concentrent sur la recherche d'une seule cible statique par un essaim de robots $N \in [10, 100]$, Les robots échangent des informations avec tous les voisins sur la détection de cible en transmettant des messages courts pour indiquer si la cible est détectée dans un rayon de RC = 10 cm, la cible est représentée par un cercle. Ils ont réalisé une simulation Multi-Agents simple qui résume les détails physiques des Kilobots et se concentre sur le modèle de mouvement collectif. Cette simulation traite les agents comme des particules sans dimension qui peuvent se déplacer à une vitesse constante et peuvent communiquer avec tous les voisins dans le rayon r_c, tout comme les Kilobots. Contrairement aux Kilobots, cependant, les agents ne se heurtent pas et peuvent changer de direction instantanément. Enfin, les agents peuvent entrer en collision avec les murs, s'ils sont présents, de sorte que le vecteur de mouvement orthogonal au mur est annulé et que seule la composante parallèle est exécutée.

Les expériences réalisées avec les Kilobots ont été réalisé seulement dans l'espace délimité. Les auteurs construisent une arène carrée avec L = 90 cm ou les robots et la cible sont distribuées uniformément. La cible est représentée par un Kilobot immobile programmé pour diffuser en continu et fréquemment un code d'identification de cible id_t. Du qu'un robot entre dans la zone de communication du Kilobot cible et reçoit le code id_t pour la première fois, il stocke un horodatage à utiliser pour les statistiques de temps de premier passage moyen. De plus, les robots communiquent entre eux et partagent des informations sur la découverte de la cible en diffusant un code d'identification de découverte id_d, soit lorsqu'un robot a découvert la cible par lui-même, soit lorsqu'un robot a reçu les informations d'un autre robot. Lorsqu'un nombre suffisant de robots a découvert la cible ou après une heure d'expérimentation.

L'analyse de différentes marches aléatoires dans un espace borné montre que la marche aléatoire cohérente (CRW) avec une persistance relativement élevée est la meilleure stratégie à adopter. Le Levy Walk (LW) a de moins bonnes performances en raison de l'effet des collisions avec les murs (ou d'autres robots) qui entraînent une mauvaise performance globale. Dans le scénario spatial illimité, la meilleure stratégie est plutôt la LW, bien qu'une certaine corrélation dans le mouvement fournisse également un avantage. La distance entre la cible et la place centrale peut avoir une grande influence sur les performances du système. De plus, le LW fournit une évolutivité au système en terme de diffusion d'informations.

16) Distributed Deterministic Spiral Search Algorithm (DDSA) [Matthew et al., 2016]

L'algorithme de recherche en spirale déterministe distribuée (Distributed Deterministic Spiral Search Algorithm- DDSA), vise à éliminer le caractère aléatoire inhérent aux approches aléatoires en faisant en sorte que les agents se déploient vers l'extérieur à partir du dépôt de collecte selon un modèle spiral. Les agents DDSA rapportent les ressources au dépôt dès qu'elles ont été trouvées, puis retournent toujours à l'emplacement où ils ont trouvé la dernière cible. C'est pour que la spirale de recherche puisse être rejointe au moment où elle a été interrompue, Le processus de retour à l'emplacement de la dernière cible trouvée s'appelle fidélité de site et constitue une stratégie de recherche courante chez les fourmis. DDSA exige que chaque robot connaisse la taille de l'essaim et son index, les robots s'éloignent de l'emplacement central et effectuer un mouvement Nord (N), Est (E), Sud (S), Ouest (W), ces mouvements son symétrique c'est-à-dire $D_N=D_S$ et $D_E=D_W$ (où D_H est la distance parcourue par le robot actuel dans une direction cardinale donnée, H pour un nombre de circuits particulier, c). Le nombre de circuits est le nombre de fois qu'un robot a effectué un mouvement dans les quatre directions.

DDSA a montré de meilleures performances par rapport au central Place Foraging Algorithm (CPFA) en termes de temps de recherche et collecte. Cependant, le CPFA a surperformé le DDSA en termes de nombre d'objets collectés dans le temps pour les gros essaims avec plus de 15 robots, parce que DDSA souffert de collisions de robots dans des environnements plus encombrés.

17) Collective Levy Walk for Efficient Exploration in Unknown Environments [Yara et al., 2018].

Les auteurs dans **[Yara et al., 2018]** ont proposé un algorithme Collective Lévy Walk (CLW) pour les essaims de robots qui exploitait la communication locale entre les robots pour générer une marche collective de Lévy. L'algorithme CLW priorise les pas les plus longues dans les trajectoires des robots en exploitant les informations échangées entre les robots. L'algorithme CLW est décrit par un automate

fini déterministe. Chaque robot démarre dans l'état de marche pour explorer l'environnement inconnu. Les robots se déplacent à une vitesse linéaire fixe et échantillonnent la durée de leur prochaine étape (T_L) à partir d'une distribution de Lévy. Lorsque l'intervalle T_L est terminé, le robot passe à l'état de rotation et tourne à une vitesse angulaire constante T_U, avec T_U échantillonné à partir d'une distribution uniforme. Chaque fois que le robot ambulant détecte un obstacle à l'aide de ses capteurs de proximité, il quitte immédiatement l'état de marche et commence à exécuter un comportement d'évitement de collision. Dans cet état, le robot tourne avec un angle déterminé en fonction de la distance de l'obstacle. Lorsque tous les obstacles ont été évité, le robot passe à nouveau à l'état de marche et échantillonne un nouvel intervalle de temps T_L pour passer à l'étape suivante de la marche Lévy.

La partie principale de l'algorithme CLW réside dans l'exploitation de la communication entre les robots pour générer une marche collective de Lévy. Le robot diffuse son intervalle de temps échantillonné T_L à ses voisins locaux, ces robots sont les robots dans sa portée de communication et secteur de vision. L'étape est classée comme courte ou longue en fonction d'un seuil prédéfini F.

Les auteurs ont effectué un ensemble de simulations en utilisant le simulateur ARGoS. Une arène de 20 × 20 m² est implémentée comme un espace illimité : le robot qui atteint un côté de l'arène rentrera du côté opposé sans interrompre l'exécution de l'étape actuelle.

Les résultats de simulation sont calculés en moyenne sur 30 exécutions, chaque exécution dure 5000 pas de temps. L'exposant α est réglé sur 1, la portée de communication du robot est réglée sur 1,35 m et la vitesse linéaire sur 5 m / s. Le seuil de pas est réglé sur F = 1,53 m (0,17 le diamètre du robot simulé). Les résultats montrent la capacité du CLW à générer une telle trajectoire collective également pour de plus grandes tailles d'essaims.

Comparaison qualitative des travaux reliés

Nous avons comparé les travaux reliés dans le **Tableau 2.1**. La comparaison est faite à la base de plusieurs critères que nous avons juges importants, ces critères concernent les caractéristiques des robots, des cibles, de l'environnement, des stratégies suivies par les robots et la nature de test ou expérimentations réalisés. Un croix dans la case veux dire que la caractéristique existe dans cette article.

Axe 1	Axe 2	Axe 3	Les références																
			(Sanchez et al., 2018)	(Pang et al., 2019)	(Yara et al., 2018)	(Santos et al., 2018).	(Palmieri et al., 2018)	(Palmieri et al., 2017)	(De Rango et al., 2018)	(Palmieri et al., 2015)	(Palmieri et al., 2016)	(Palmieri et al., 2015)	(Sutantyo et al., 2013)	(Katada et al., 2016)	(Abuomar et Al-Aubidy.2018)	(huang et al., 2019)	(Suárez et Iglesias, 2017)	(Dimidov et al., 2016)	(Matthew et al., 2016)
Environnement	**Complexité**	**Avec obstacle**			X	X	X	X	X						X	X	X		
		Sans obstacle	X	X						X	X	X	X	X				X	X
Objets	**Distribution**	**Aléatoire**	-	X	X		X	X	X	X	X	X		X	X	X	X	X	X
		Partiellement regroupés																	X
		Entièrement regroupés																	X
	Nature	**Statique**						X	X		X	X		X	X	X	X		
		Dynamique					X												
Robots	**position**	**central**		X															
		Aléatoire		X										Position prédéfini	X				
	énergie	**Limitée**		X			X	X					X	X		X	X	X	
		Illimitée	X		X	X			X	X	X	X			X				X
	perception	**Illimitée**																	
		Limité	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X
	mémorisation	**Oui**														X			
		Non					X	X	X	X	X		X				X	X	X
	Homogénéité	**Oui**	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X
		Non														X			
Stratégies	**communication**	**Directe**	X		X				X(Connexion sans fil)					X(Connexion sans fil	X(Connexion sans fil)			X	
		Indirecte					X(phermon) (atra	X(phermon)	X(phermon)	X(phermon) (atraction)	X(phermon) (atraction)	X(phermon) (atraction)	X(atraction)				X		X(phermon)

							ction)												
		inspiration					ACO|FA	ACO|FA|PSO|ABC	ACO	ACO|FA	ACO|FA	ACO|FA	FA						ACO
	exploration	**Aléatoire**	X	X	X(levy)									X(levy)		X	X	X	
		Stratégique	X	X		X	X(pheromon)	X(pheromon)	X(pheromon)	X(pheromon)	X(pheromon)	X(pheromon)	X(levy)		X				X
		inspiration	PSO	LF|BM		FA	ACO	ACO	ACO	ACO	ACO	ACO			dragonfly				DDSA
	recrutement	**Oui**	X				X	X	X	X	X	X				X			
		Non		X									X		X			X	
		Inspiration					FA	FA|PSO|ABC	ACO	ACO|FA	FA	ACO|FA				bat			
Simulation	**Monde réelle**	**Expérimentation réel**		X(e-puck)									X	X			X	X	
	Plateforme	**Argos**			X														X
		Player/ Stage																	
		Webots		X															
		Gazebo	X																
		basé sur Java					X	X	X	X	X	X							
		Netlogo											X						
		SIMROBOT													X				
		Mona Robot														X			
		Unity 5															X		
		V-REP(ROS)				X													

Tableau 2.1 : Comparaison qualitative des travaux reliés.

D'après le ***Tableau 2.1,*** on peut faire plusieurs constatations :

- La majorité des travaux ont utilisé un environnement sans obstacles ;
- La plupart des travaux utilisent des cibles statiques avec distribution aléatoire ;
- Sauf quelques travaux qui ont fixé la position des robots, comme **[Pang et al.,2019]** qui a suivi les deux stratégies centrale et aléatoire, **[Katada et al.,2016]** où les robots ont une position prédéfinie et **[Abuomar et Al-Aubidy.,2018]** qui a suivi la stratégie aléatoire. La moitié des travaux considèrent une énergie illimitée des robots et l'autre moitié considèrent une énergie limitée ;
- La majorité des travaux considèrent une perception des robots limitée ;
- Tous les travaux utilisent des robots homogènes et la plupart des travaux utilisent des robots sans mémoire ;
- La plupart des travaux ont utilisé des algorithmes bio-inspirés comme stratégie d'exploration et des comportements de communication directs et indirects pour les robots et aussi la majorité utilisent le travail collaboratif ;
- Les expériences ont été testé soit dans le monde réel comme **[Yara et al, 2018]** qui a utilisé les robots e-puck et **[Matthew et al.,2016]** ou des simulations dans des plateformes comme ARGoS, Player/Stage et Gazbo…etc et des autres plateformes auto développé.
- Les travaux utilisant l'algorithme FA utilise sont comportement lumineux attractif.

IV Conclusion

Nous avons commencé ce chapitre par présenter quelques définitions du problème d'exploration Multi-Robots. Puis, nous avons présenté une synthèse des travaux reliés qui se termine par un tableau comparant les différents travaux.

Dans le chapitre suivant, nous proposons une solution au problème d'exploration Multi-Robots, toute en inspirons des algorithmes issus de l'intelligence en essaim.

Chapitre 3 : Conception

I. Introduction

Nous considérons dans ce travail le problème d'exploration Multi-Robots. Dans lequel un ensemble de robots doivent rechercher des objets dans leur environnement. Pour une recherche plus efficace, les robots doivent repartir l'environnement entre eux implicitement et de se disperser ce qui leur donne la chance de localiser plus d'objets. L'exploration dans ce travail constitue une phase préliminaire pour la recherche des targets.

L'algorithme FA a été toujours utilisé pour attirer les robots au robot avec l'intensité de lumière la plus élevée. Dans ce travail, nous suivant la logique contraire, c'est-à-dire que nous allons utiliser l'algorithme FA pour repousser les robots les uns des autres lorsqu'un robot détecte un objet. Le robot commence par explorer son environnement par une marche aléatoire simple (random bounce) ou stratégique (global lawnmawer), lorsqu'il détecte un objet il envoie une lumière a son voisinage, et commence à exploiter localement cette région (supposée riche) en utilisant une marche en spirale, tandis que les autres robots s'éloignent de la région courante vers d'autres, ce qui entrainera une large dispersion des robots. Nous avons proposé deux algorithmes Random Bounce Firefly (RBF) et Global Lawnmower Firefly (GLF) utilisant tous les deux la logique contraire de la lumière des lucioles.

Dans ce chapitre, nous commençons par description de la problématique, les objectifs et la proposition. Ensuite, nous présentons les algorithmes reliés (algorithme FA, random bounce et global lawnmawer). Puis, nous présentons le pseudo code des algorithmes proposes (RBF et GLF) et la machine d'état associée.

II. Problématique, objectifs et proposition

L'exploration Multi-Robots consiste en un ensemble de robots qui doivent rechercher toute les zones de l'environnement dans l'objectif de localiser un ensemble d'objets. Dans les environnements inconnus, la marche aléatoire fait une bonne alternative qui donne la chance de localiser plus d'objets, mais elle peut être coûteuse en temps du fait que les robots peuvent rechercher des zones déjà explorées de façon répétitive. De plus, les robots peuvent ne pas atteindre les zones les plus éloignées et donc la chance de

localiser plus d'objets sera minimale. La large dispersion des robots permet d'explorer plus de zones et donne la chance de trouver plus d'objets. La répartition des zones entre robots est une autre alternative qui permet l'exploration rapide des environnements, mais c'est difficile de l'assurer dans les essaims de robots qui manquent de communication directe.

Nous avons fixé trois objectifs principaux :

1. Assurer une large dispersion des robots dans leur environnement pour augmenter le nombre d'objets trouvés ;
2. Diviser l'espace de recherche entre les robots implicitement à travers une communication indirecte ;
3. Assurer une exploration locale efficace ;

Pour répondre aux objectifs fixés précédemment, nous avons jugé intéressant d'utiliser des algorithmes issus de l'intelligence en essaim. Dans des travaux antérieurs **[Idiri, 2018]**, et comme toute la littérature, nous avons utilisé l'algorithme FA pour explorer des environnements inconnus. Chaque robot a été doté d'une lumière qui lui permet de communiquer avec les autres robots. Cette lumière a été utilisée principalement pour attirer les robots à certaines régions pour une exploitation collective. Par contre, dans ce travail nous allons utiliser l'algorithme FA pour éloigner les robots les uns des autres et donc d'assurer une sorte de division implicite de l'environnement entre les robots (la répulsion se fait seulement quand un objet est localisé). Ce comportement de répulsion en combinaison avec la marche aléatoire permet aux robots d'explorer des régions plus éloignées assurant une large dispersion des robots et localisant plus de targets.

Nous avons utilisé la marche aléatoire Random Bounce (RB) et la marche Global Lawnmower (GL) en hybridation avec l'algorithme FA.

III. Les Algorithmes reliés

Les algorithmes que nous avons utilisés dans notre proposition sont : la marche aléatoire Random Bounce (RB) et la marche Global Lawnmower(GL) et l'algorithme de Firefly (FA) et Spiral. Nous donnerons dans cette section, une description plus au moins détaillée de ces algorithmes :

III.1. ALGORITHME FIREFLY (FA)

C'est une technique d'optimisation méta-heuristique très connue, développé par Xin-She Yang en 2009 (**Yang,2010)**, et inspirée du modèle de clignotement et des caractéristiques des lucioles naturelles.

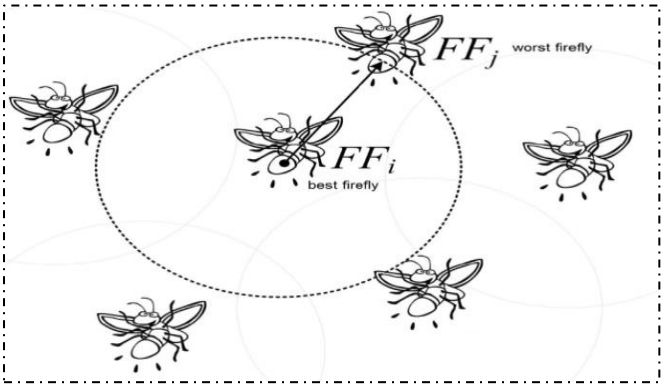


Figure 3.1 : comportement d'attraction des lucioles **[Ali et al.,2019]**

L'algorithme suit les trois hypothèses suivantes **[Yang ,2009] :**

1. Toutes les lucioles sont unisexes.
2. L'attractivité des lucioles est proportionnelle à la luminosité.
3. La luminosité de la lumière clignotante peut être considérée comme une fonction objective qui devra être optimisé.

Dans FA, la luminosité (I) et l'attractivité (β) jouent un rôle important. En général, pour un problème maximum, la brillance d'une luciole est positivement relative à la valeur de la fonction objective, et pour un problème minimum, elles sont négativement relatives entre elles.

Dans un milieu donné, l'intensité lumineuse est définie par l'équation :

$$\mathrm{I} = I_0 e^{-\gamma \mathrm{r}} \quad \text{.....(1)}$$

Où : $\boldsymbol{\gamma}$ le coefficient d'absorption, r la distance.

Avant de définir l'attractivité (à savoir β), tout d'abord, la distance euclidienne entre deux lucioles i et j doit être calculée comme suit :

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad \text{.....(2)}$$

L'attractivité d'une luciole varie selon l'intensité de la lumière des lucioles adjacentes, elle est définie par :

$$\beta = \beta_0 e^{-\gamma \mathrm{r}^2} \quad \text{.....(3)}$$

Où : $\boldsymbol{\beta 0}$ est l'attractivité quand r=0.

Supposons que $\mathbf{x_i}$ soit attiré par $\mathbf{x_j}$, et qu'en suite, $\mathbf{x_i}$ se déplace comme suit :

$$x_i = x_i + \beta_0 e^{-\gamma r^2_{ij}} \left(x_j - x_i\right) + a(rand - \frac{1}{2}) \text{ .....(4)}$$

III.2. Algorithme Random Bounce (RB)

Dans l'algorithme RB, chaque robot est supposé avoir la capacité de détecter les limites de l'environnement, par le biais de certaines informations telles que les capteurs de proximité qui peuvent détecter les obstacles dans l'environnement ou les murs de l'environnement. De plus, chaque robot est supposé avoir la capacité de détecter les autres robots dans l'environnement (en utilisant les mêmes capteurs de proximité). Chaque robot avance à une vitesse fixe jusqu'à ce que la limite de l'environnement, un autre robot ou obstacle soit détecté. À ce moment, le robot choisisse un angle aléatoire face à un bord différent de l'environnement et en se dirigeant en ligne droite dans cette direction, cela se répète aussi longtemps que le robot explore **[Isaacs et al., 2020].**

Le robot tournera aléatoirement vers une nouvelle direction selon la formule suivante,

$$\theta_{new} = \theta + \text{n} \text{.....(5)}$$

Où **θ** est la direction du robot au moment de la détection, ***n*** est une variable aléatoire uniforme, et la distribution est décidée par quel côté du robot a rencontré l'obstacle déclenchant la rotation. Si le robot détecte quelque chose sur le côté gauche $n \sim U\ (\pi/4,3\pi/4)$, sur le côté droit $n \sim U\ (-3\pi/4,-\pi/4)$, et au centre $n \sim U\ (3\pi/4,5\pi/4)$.

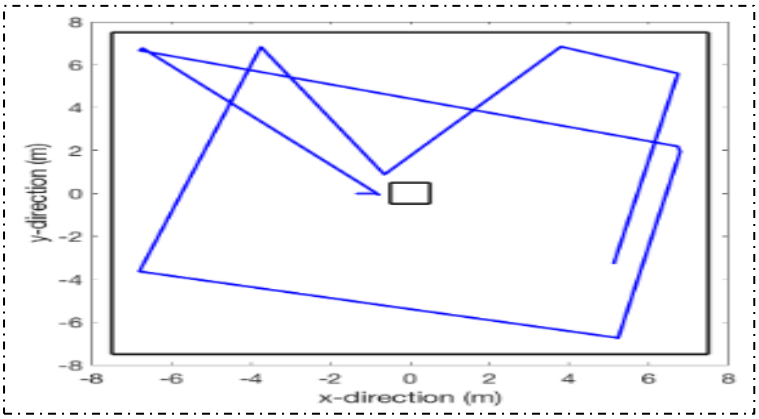


Figure 3.2 : La trajectoire prise par un seul robot sous l'algorithme de recherche par rebond aléatoire (RB) **[Isaacs et al., 2020]**

III.3. Algorithme global lawn-mower (GL)

La méthode de recherche de global lawn-mower, illustrée par la **Figure 3.3**, est l'un des modèles de recherche les plus utilisés en raison de sa simplicité et de sa garantie de continuité et de douceur du chemin **[Ousingsawat et Earl., 2007]**.

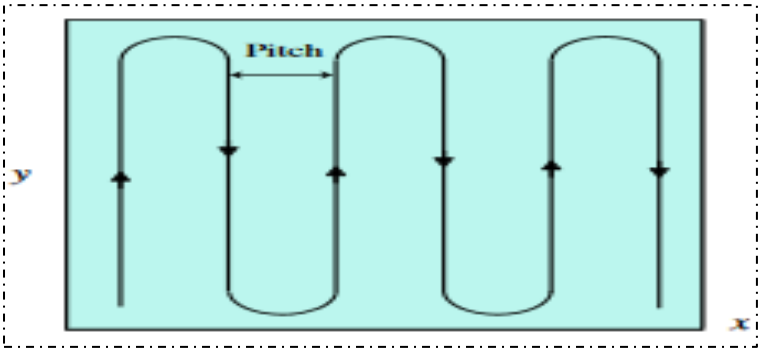


Figure 3.3 : global lawn-mower **[Ousingsawat et Earl, 2007]**

Le motif se compose de deux mouvements majeurs : (1) chemin droit ; (2) chemin en demi-cercle. Le '*Pitche*' ou 'Pas' dans la **Figure 3.3** indique l'espacement entre deux trajectoires droites consécutives. Les coins et les bordures sont souvent manquées et négligées en raison de la rotation. En fait, de nombreux types de recherche souffrent de cette dépendance.

III.4. Algorithme Spiral carré

L'idée derrière une recherche en spirale carrée est simple : avancer une fois, tourner de 90 degrés dans le sens inverse des aiguilles d'une montre, répéter, avancer deux fois, tourner à 90 degrés, répéter, avancer trois fois… etc (voir **Figure 3.4**).

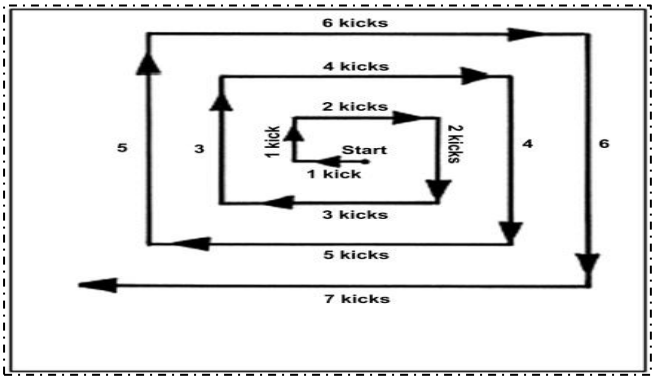


Figure 3.4: Spiral carré **[Arbeit, 2014]**.

Il fournit une approche simple pour rechercher le long d'un chemin spécifié qui est utile lorsque l'itinéraire prévu de la cible de recherche est connu. Il fournit une couverture rapide et raisonnablement complète de la zone le long d'un chemin. Les entrées du modèle sont la ligne de trace spécifiée comme une série de coordonnées, et l'espacement des traces **[Arbeit, 2014]**.

IV. Algorithmes proposés

Nous présentons dans cette partie, deux algorithme GLF et RBF proposé. Commençons par montrer la description des algorithmes RBF et GLF, puis nous présentons le pseudo code de l'algorithme GLF et RBF, par la suite nous présentons la machine d'état associés aux robots GLF et RBF.

IV.1 Description des algorithmes RBF et GLF

Nous considérons les règles suivantes :

1. **Gestion de l'exploration globale :** pour l'algorithme RBF on utilise la marche RB et pour l'algorithme GLF on utilise la marche GL décrites comme suit :
   1. *Random bounce :* Chaque robot avance simplement à une vitesse fixe jusqu'à ce que la limite de l'environnement, un autre robot ou obstacle soit détecté. À ce moment, il choisissant un angle aléatoire face à un bord différent de l'environnement et en se dirigeant en ligne droite dans cette direction, cela se répète aussi longtemps que le robot recherche. Le robot tournera aléatoirement vers une nouvelle direction selon l'équation **(5)**.
   2. *Global lawn-mower :* cette méthode est composée deux mouvements principaux :
      1) Chaque robot avance simplement à un nombre des pas à une vitesse fixe. Le nombre de pas est généré par la loi de Lévy selon l'équation :

$$\mathrm{PL} = l_0 \left(\frac{1}{\mathrm{U}^{1/\beta}} - 1\right) \quad \text{.....(6)}$$

PL : La longueur de pas générer par la fonction Lévy. $\boldsymbol{l_0}$: Variable d'échelle $\boldsymbol{\gamma}$>0,
U : Une valeur réelle aléatoire dans l'intervalle [0,1].

      2) A la fin de la première étape, le robot tourne à un angle de 90 degrés, puis le robot avance simplement d'un nombre fixe de pas, puis tourne à un angle de 90 degrés et revient au premier pas.

2. **Gestion de l'exploration locale :** le robot utilise une marche en spirale carrée (voir section III.4).
3. **Gestion de l'intensité lumineuse :** l'intensité lumineuse *I* est gérée en utilisant les deux règles suivantes :
   - Augmenter l'intensité lumineuse à chaque fois une cible est trouvée selon l'équation :

$$I_{t+1} = I_t \times T \quad \text{.....(7)}$$

   $\boldsymbol{I_t}$: L'intensité de lumière à l'instant t, **T** : la valeur d'augmentation de l'intensité.
   - Diffuser l'intensité lumineuse par rapport à la distance à partir de la source selon l'équation :

$$I_{(r)} = I_t \times \mathrm{K} \text{.....} (\mathbf{8})$$

$\boldsymbol{I_r}$: L'intensité de lumière à distance r, **K** : la valeur d'augmentation la distance.

4. **Gestion de l'évitement de la lumière :** Pour éviter la lumière détectée, le robot utilise la règle du mouvement selon l'équation :

$$x_{i+1} = x_i - \beta_0 e^{-\gamma r^2_{ij}} \left(x_j - x_i\right) + a(rand - \tfrac{1}{2}) \text{.....} (\mathbf{9})$$

$\boldsymbol{\beta_0}$: Attraction, $\boldsymbol{x_i}$: La position du robot i.

Le robot s'éloignera de la lumière détectée jusqu'à ce qu'il n'y ait plus de lumière détectée.

5. **Gestion de la vitesse du robot :** la vitesse des robots est augmentée lorsqu'une lumière est détectée selon l'équation :

$$v_{i+1} = v_i \times z_i \text{.....} (\mathbf{10})$$

$\boldsymbol{v_i}$: La vélocité (vitesse) du robot a l'instant , $\boldsymbol{z_i}$: Coefficient d'inertie du robot i.

D'après Yang et Deb dans **(Yang et Deb, 10)**, nous avons initialisé β_0=1. Aussi le secteur de vision $\boldsymbol{S_r}$ est fixée à 0,3 m ;

Symboles	Signification
θ	la direction du robot au moment de la détection.
n	est une variable aléatoire uniforme.
PL	La longueur de pas générer par la fonction Lévy.
l_0	Variable d'échelle $\boldsymbol{\gamma}$>0.
U	Une valeur réelle aléatoire dans l'intervalle [0,1].
I_t	L'intensité de lumière à l'instant t.
T	la valeur d'augmentation de l'intensité
$I_{(r)}$	L'intensité de lumière à distance r
K	la valeur d'augmentation de la distance
β_0	Attraction
x_i	La position du robot i
v_i	La vélocité (vitesse) du robot a l'instant t
z_i	Coefficient d'inertie du robot i
S_r	Le secteur de détection

Tableau 3.1: symboles adoptés dans le problème.

IV.2 Pseudos codes des algorithmes RBF et GLF

Les algorithmes RBF et GLF partagent la même définition des comportements : *Détruire_cible*, *Eviter_obstacle*, et *Eviter_lumière*. Les seuls comportements qui diffères sont ceux de l'exploration. Nous présentons dans ce qui suit les pseudos-codes de ces comportements.

Algorithme III. 1.1 : Exploration avec Random Bounce

1. **Tant que** (∄ obstacle ou cible ou lumière) **faire**
2. Explorer globalement en utilisant la **RB**
3. **Fin Tant que**
4. **Si** (∃ obstacle) **alors** Aller à ***Eviter_obstacle***
5. **Si** (∃ cible) **alors** Aller à ***Détruire_Cible***
6. **Sinon Si** (∃ lumière) **alors** Aller à ***Eviter_Lumière***
7. **Fin Si**

Algorithme III. 1.2 : Exploration avec Global Lawn-mower

1. **Tant que** (∄ obstacle ou cible ou lumière) **faire**
2. Explorer globalement en utilisant **GL**
3. **Fin Tant que**
4. **Si** (∃ obstacle) **alors** Aller à ***Eviter_obstacle***
5. **Si** (∃ cible) **alors** Aller à ***Détruire_Cible***
6. **Sinon Si** (∃ lumière) **alors** Aller à ***Eviter_Lumière***
7. **Fin Si**

Algorithme III. 2 : *Détruire_Cible*

1. **Tant que** (nb_tours < 5) **faire**
2. Explorer localement en utilisant la marche en spiral carré
3. **Si** (∃ cible) **alors**
4. Mettre à jour le nb_cible ;
5. Mettre à jour le nb_cible_distruct ;
6. **Fin Si**
7. **Si** (∃ obstacle) **alors** Aller à ***Eviter_obstacle***
8. **Fin Tant que**
9. Aller à Exploration (Algorithme III.1.1 ou Algorithme III.1.2)

Algorithme III. 3 : *Eviter_obstacle*

1. **Récupérer** les lectures de tous les capteurs de proximité
2. **Calculer** la valeur V de la lecture maximale Lmax
3. **Si** (V > 0) **alors**

4. **Calculer** l'angle A de Lmax
5. **Si** A montre que l'obstacle est devant le robot **alors** Marche en arrière
6. **Sinon Si** A montre que l'obstacle est dans le côté gauche **alors** Déplacer à droite et choisissez un angle aléatoire
7. **Sinon** Déplacer à gauche et choisissez un angle aléatoire
8. **Fin Si**

Algorithme III. 4 : *Eviter_Lumière*

1. **Tant que** (la lumière est détectée) **faire**
2. Éloignez-vous de la source lumineuse avec double vélocité en utilisant l'équation (10)
3. **Si** (∃ obstacle) **alors** Aller à ***Eviter_obstacle***
4. **Si** (∃ cible) **alors** Aller à ***Eviter_obstacle***
5. **Fin Tant que**
6. **Si** (lumière n'est pas détectée) **alors** Aller à Exploration (Algorithme III.1.1 ou Algorithme III.1.2)

IV.3 Machine d'état des robots

La machine d'état représentant les deux algorithmes est donnée par la **Figure3.1** Où ***NB_tours*** est une valeur fixe et prédéfinie pour le nombre de tours que le robot effectue sans trouver de cible et lumière sa valeur est déterminée par l'équation **(7)**, où l'intensité de la lumière représente la distance à partir de laquelle le robot évite la source de lumière.

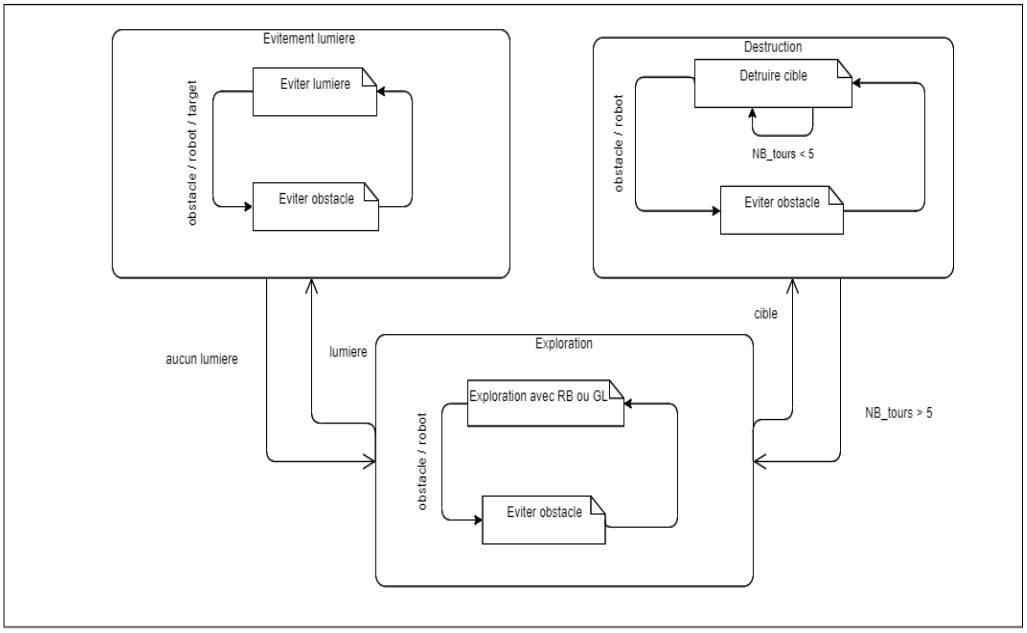


Figure 3.5 : Machine d'état des algorithmes RBF et GLF

- **Exploration (recherche globale) :** nous avons tester dans cette phase deux méthodes d'exploration : (1) la première méthode utilise la stratégie Random Bounce et (2) la deuxième méthode utilise la stratégie Global Lawnmwer :

  Il y a deux conditions principales permettent à un robot de passer d'un état à un autre :

  1. Si le robot détecte une cible, il passe à l'état de **Destruction**, Cela changera la façon de marcher vers le spiral carré pour détruire le plus grand nombre de cibles voisines. Lorsque le robot effectue un certain nombre de rotations (prédéfinies) sans trouver de cible, il passe à l'état ***Exploration avec Global Lawn-mower*** ou ***exploration avec random bounce.***
  2. Si le robot détecte une source de lumière, il entrera dans un état d'***Evitement_Lumière*** loin de la lumière détectée.

- **Destruction (recherche locale) :** le robot utilise la méthode de marche en spirale. Cela permet d'effectuer une recherche locale approfondie de toutes les zones voisines. Chaque fois qu'une cible est détectée, le robot augmente l'intensité de la lumière en utilisant l'équation **(7)** pour diffuser la lumière pour éloigner les robots qui sont à leur portée ou qui les approchent. Lorsque le robot effectue un certain nombre de tours sans trouver de cible, il passe à l'***Exploration avec Global Lawn-mower*** ou *l'**exploration avec random bounce***.

- **Evitement_lumière :** Lorsqu'un robot explorateur détecte une lumière, il change d'état vers *éviter_lumière* et se déplace contre la source de la lumière en utilisant l'équation **(9)** jusqu'à ce qu'il ne détecte plus de lumière. Tout en évitant la lumière, le robot trouve une cible qu'il évite et la considère comme un obstacle (la priorité absolue est d'éviter la lumière). Lorsque le robot ne détecte aucune lumière, il retourne à l'état d'exploration.

- **Eviter_obstacle :** Lorsqu'un robot explorateur détecte un obstacle, il vérifie s'il peut avancer en toute sécurité ou s'il doit tourner à un angle opposé à l'angle de lecture en utilisant l'équation (5).

V. Conclusion

Plusieurs applications de robotique mobile comme le nettoyage, le sauvetage, le désarmement, le foraging…etc, se base d'une partie sur la phase d'exploration. Un tel comportement de base a été résolu depuis les années par des algorithmes issus de l'intelligence en essaim (ACO, PSO, FA, ABC…etc). Nous avons tenté dans ce mémoire de proposer une solution hybride qui améliore la recherche aléatoire

par l'utilisation de lumière des lucioles par pour attirer mais plutôt pour repousser les robots a des zones distinctes.

Nous avons donc amélioré la marche aléatoire random bounce par utilisation de la lumière en cas des cibles sont localisées, et nous avons aussi améliorer la marche global lawnmower par utilisation de la lumière. Nous avons utilisé la lumière pour repousser les robots a d'autres zones dans le but d'explorer le maximum possible de zones et donc de localiser plus de cibles.

Chapitre 4 : Implémentation et Simulations

I Introduction

Dans ce chapitre nous cherchons à tester quantitativement les performances des deux algorithmes améliorés : Random Bounce Firefly (RBF), Global Lawn-mower Firefly (GLF). Nous avons implémenté les deux algorithmes sous la plateforme de robotique mobile ARGoS. Pour tester l'effet des améliorations faites nous avons réalisé plusieurs simulations dans des configurations environnementales différentes.

Nous commençons ce chapitre, par présenter la plateforme ARGoS. Par la suite nous décrivons le type et les caractéristiques des robots utilisés. Nous proposons apres un ensemble de scenarios de simulations. Puis, nous présentons et discutons les résultats obtenus par les algorithmes proposés. Nous terminerons le chapitre par une conclusion.

II Environnement de développement

ARGoS (**A**utonomous **R**obots **G**o **S**warming) **(Pinciroli, 2012)**, est un simulateur de robots crée par Carlo Pinciroli pendant le projet Swarmanoid. Il est développé dans le laboratoire IRIDIA. Le but de ce projet était de développer un outil flexible et efficace afin de simuler des expériences complexes impliquant des essaims de robots de différents types. ARGoS s'inscrit dans une optique aussi flexible qu'efficace. Il se veut en effet être flexible grâce à la modularité introduite dans celui-ci : ses modules, vus comme des plugins peuvent être configurés, étendus, inters changés suivant les nécessitées des différentes expériences. Différentes implémentations des mêmes composants peuvent même exister afin d'obtenir différents niveaux de précisions, et leurs coûts correspondants. D'un autre côté, l'efficacité se retrouve dans le fait de ne calculer que les composants nécessaires, tout en créant une architecture parallèle afin de pouvoir faire usage des processeurs multi-coeurs. ARGoS est un simulateur disponible de manière libre et gratuite et est utilisé par de plus en plus de laboratoires à travers le monde.

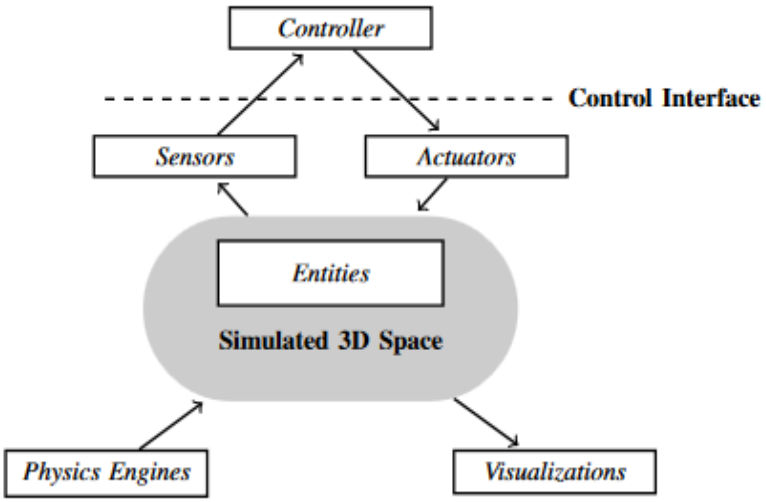


Figure 4.1 : L'architecture du simulateur argos.

L'architecture d'ARGOS est donnée par **la Figure 4.1.** Le contrôleur est le module principal qui contient l'algorithme du robot. Les autres cases sont des modules qui peuvent être modifie par l'utilisateur. Une explication des différents modules de l'architecture d'ARGOS est donnée dans ce qui suit :

- **Espace 3D simule (Simulated 3D space) :** c'est une collection de structures de données qui contient l'état complet de la simulation. Cet état d'information contient la position et l'orientation de chaque objet tel que les obstacles, les cibles ou les robots.
- **Entités (Entities) :** les données sont organisées dans des éléments de base appelés entités. Dans ARGOS, ils existent plusieurs types d'entités, et l'utilisateur peut personnaliser ou ajouter de nouveaux si nécessaires.
- **Capteurs et Actionneurs (Sensors and Actuators) :** Les capteurs et les Actionneurs sont des plug-ins qui accèdent à l'état de l'espace 3D. les capteurs peuvent accéder et lire uniquement l'état de l'espace, tandis que les Actionneurs sont autorisés à le modifier.
- **Moteur physique (physical engine) :** les moteurs physiques sont des modules qui mettent à jour l'état des entités incorporées dans l'espace.
- **Visualisations** : ce sont des modules qui lisent l'état de l'espace 3D simule et émettent une représentation de celui-ci dans la fenêtre de simulation.
- **Contrôleur (Controller) :** les contrôleurs de robot sont des plug-ins qui contiennent les codes qui contrôlent les comportements des robots qui sont actuellement écrits en C++.

- **Fonctions de boucle (Loop Functions)** : il est utilisé pour permettre à l'utilisateur de contrôler et de personnaliser les événements qui se déroulent pendant la simulation, et d'ajouter des nouvelles fonctionnalités. Par exemple, ajoutes, supprimes où changer l'état interne des objets ou des entités dans l'environnement à cause des actions des robots. La fonction de boucle est exécutée avant et/ou âpres chaque étape de simulation. Elle permette d'accéder et de modifier toute la simulation. De cette manière, l'utilisateur peut collecter des chiffres et stocker des données complexes pour une analyse ultérieure.

III Modélisation des composants de notre système

Notre système Multi-Robots se compose de cinq composants principaux : l'environnement, les robots, les cibles, les obstacles et la lumière. Nous montrons dans ce qui suit la modélisation proposée pour ces composantes.

1. L'environnement est modélisé par un espace 3D continu ;
2. Les robots sont des Footbots. Le Foot-bot est une configuration particulière de modules basée sur la plateforme robotique marXbot. Cette configuration comprend un module CPU, un scanner de distance, un module d'auto-assemblage et un module de base (mobilité et batterie) ;
3. La zone de lumière d'un robot est représentée par un cercle jaune fané ;
4. Les cibles sont représentées par un cylindre gris 3D de taille 2.5cm X 10cm ;
5. Les obstacles sont représentés par différentes formes de boites grises 3D de tailles différentes.

III.1 Caractéristiques du Foot-bot utilisé dans nos algorithmes

Le Footbot est un robot mobile différentiel à deux roues d'environ 17 cm de diamètre et 29 cm de hauteur et 1,8 kg. 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.

Le footbot dans nos simulations utilise les caractéristiques suivantes :

24 capteurs IR pour la détection d'obstacles et de proximité, 24 capteurs de lumière avec un secteur de détection = 0.3m, se déplace avec une vélocité constante V= 50 cm/s, aucune communication directe avec les autres Footbots, considère les autres robots comme des obstacles, Une entité lumineuse au-dessus de chaque Footbot pour pouvoir simuler le comportement de lucioles dans nos algorithmes.

III.2 Analyse du contrôleur des algorithmes RBF et GLF

1) Navigation :

Le Footbot se déplace dans l'environnement en utilisant une paire de roues. Dans le code de contrôle, on utilise la procédure " *set_velocity (l, r)* " pour le déplacement des robots, où "l" et "r" sont respectivement les vitesses de roue gauche et droite. Pour détecter les obstacles et collisions ou cible autour des robots nous utilisons Les capteurs de proximité et le nombre de ce capteur est par défaut 24 où chaque capteur a une portée de 0.3 m et renvoie une lecture composée d'un angle en radians et d'une valeur dans l'intervalle [0,1] : (1) Valeur : dépend de la distance entre le capteur et l'objet détecté. C'est-à-dire, la $valeur$ = 0 lorsque rien n'est détecté, et augmente de façon exponentielle à 1 lorsque les objets détectés sont plus proches. (2) Angle : est mesuré entre l'axe X local et l'angle auquel le capteur est situé sur le corps du robot. Pour détecter les sources lumineuses nous utilisons Les capteurs de lumière. Le robot a 24 capteurs de lumière par défaut, Chaque lecture de capteur est composée d'un angle en radians et d'une valeur entre 0 et 1 : (1) Angle correspond à l'endroit où le capteur est situé dans le corps du robot ; (2) valeurs = 0 signifie qu'aucune lumière été détectée par le capteur, tandis qui les valeurs > 0 signifient que la lumière a été détectée. La valeur augmente lorsque le robot se rapproche de la source lumineuse.

2) Evitement d'obstacle

À chaque étape de la simulation, Les capteurs de proximité du robot lisent les objets à proximité du robot, alors que le robot lit les valeurs des capteurs et calcule la valeur de lecture maximale entre les 24 lectures des capteurs de proximité, ce qui permet de déterminer la position et la distance de l'obstacle au corps de ce robot. Lorsque le robot détecte un obstacle (valeur> 0), il vérifie s'il peut avancer en toute sécurité ou s'il doit tourner à un angle opposé à l'angle de lecture. Si l'obstacle n'est pas entouré de lui et que la distance minimale de sécurité ne dépasse pas 45 à -45 degrés, le robot le considérera comme un danger.

IV Simulations et analyse des résultats

Dans cette section nous présentons les critères de performances utilisés, les scénarios de simulations réalises, les résultats obtenus et les discussions.

IV.1 Critères de performance

Nous avons utilise l'exploration dans le cadre de la recherche des cibles, donc nous proposons d'utilisé le critère de performance : *le nombre des cibles trouvées* dans un temps fixe (20 minutes) dans différents configurations environnementales (voir Scénarios 1, 2, 3 et 4).

IV.2 Scénarios de simulation

Nous avons réalisé les quatre scénarios de simulation, qui sont résumés dans le ***Tableau 3.2*.** Les scenarios testent l'effet de variation du nombre de robots, la taille de l'environnement, nombre de clusters et la distribution de cibles sur les performances des algorithmes proposés (RBF et GLF).

Scénarios 1 : variation du nombre des robots	**Scénarios 2 : variation de la taille de l'environnement**
Nombre des robots : 30, 40, 50, 60 ; Distribution des cibles : Clusters ; Nombre des clusters : 12 ; Nombre des cibles dans l'environnement : 910 ; Densité des obstacles : 10 % de la taille de l'environnement ; Taille de l'environnement : 120m X 120m ;	Nombre des robots : 50 ; Distribution des cibles : Clusters ; Nombre des Clusters : 12 ; Nombre des cibles : 910 ; Densité des obstacles : 10 % ; 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 des Clusters	**Scénarios 4 : la distribution des cibles**
Nombre des robots : 50 ; Distribution des cibles : Clusters ; Nombre des Clusters : 2, 4, 8, 16 ; Nombre des cibles : 1000 ; Densité des obstacles : 10 % de la taille de l'environnement ; Taille de l'environnement : 120m X 120m ;	Nombre des robots : 50 ; Distribution des cibles : Clusters, uniformes ; Nombre des Clusters : 12 ; Nombre des cibles : 600 ; Densité des obstacles : 10 % ; Taille de l'environnement : 120m X 120m ;

Tableau 4.1: Scenarios de simulations réalisés

V.3 Résultats, discussions et comparaisons

Les performances de RBF et GLF ont été teste avec les scenarios de simulations dans le **Tableau 4.1**. Nous avons comparé les deux algorithmes. Dans ce qui suit, nous présenterons les résultats obtenus par

ces deux algorithmes dans des scenarios différents, nous discuterons et analyserons les résultats obtenus. Les résultats obtenus dans les expérimentations suivantes sont la moyenne de 5 simulations.

1) Scénario 1

Dans ce scénario,L'algorithme RBF a été comparé avec l'algorithme RB basique et L'algorithme GLF a été comparé avec l'algorithme GL basique.Nous étudions l'influence du nombre de robots sur les performances des quatre algorithmes. **La Figure 4.3** montre une exécution sous ARGoS de l'algorithme RBF avec 50 robots.

N° Robots / Algorithmes	*30*	*40*	*50*	*60*
RBF	704.2	749.4	769	798.2
RB	307	360.8	417.8	441.4
GLF	600.4	615.8	633.2	713.8
GL	327.8	339.4	372	420.4

Tableau 4.2: Influence du nombre de robots sur les performances.

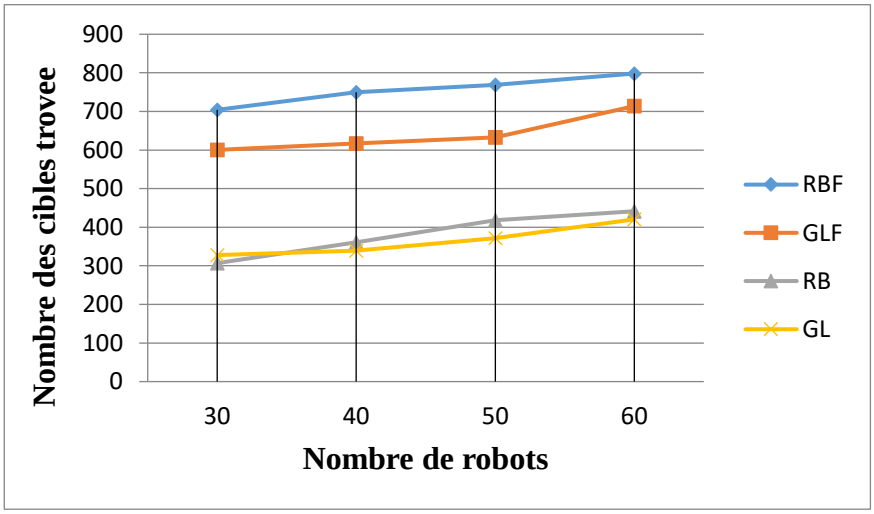


Figure 4.2 : Influence de nombre de robots sur les performances.

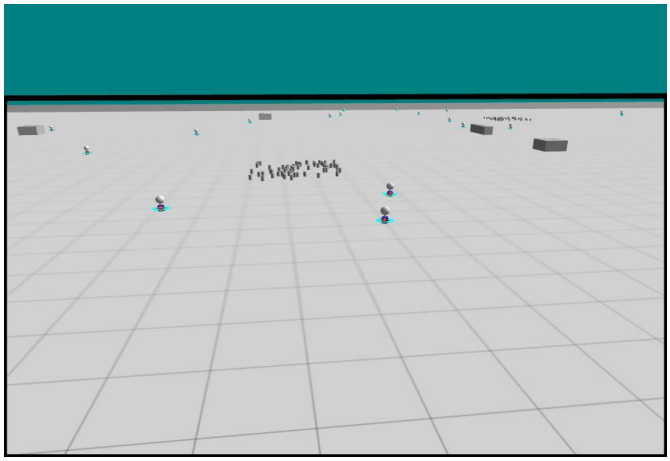


Figure 4.3 : Une exécution sous ARGoS de l'algorithme RBF avec 50 robots.

Discussion et comparaison

Le **Tableau 4.2** et la **Figure 4.2** montrent les résultats obtenus avec les algorithmes RBF et RB, et les résultats obtenus avec les algorithmes GLF et GL. Lorsque on varie le nombre des robots de 30 à 60. Le nombre de cibles trouvées augmente avec toute augmentation du nombre des robots. Nous avons observé que pour 30 robot l'algorithme RBF a trouvé 77.38% des cibles et l'algorithme RB a trouvé 33.73%, et l'algorithme GLF a trouvé 65.97% des cibles et l'algorithme GL a trouvé 36.02% a chaque augmentation de nombre de robots la performance des algorithmes va augmenter car les robots sont répartis dans la plupart des zones de l'environnement. Avec 60 robots on a observé que RBF a trouvé 87.71% des cibles et RB a trouvé 48.50%, et l'algorithme GLF a trouvé 78.43% des cibles et GL a trouvé 46.19%. On conclut que l'algorithme de RBF et GLF donne des meilleurs résultats par rapport à l'algorithme RB et GL.

2) Scénario 2

Ce scénario traite l'influence de la taille de l'environnement sur la performance des deux algorithmes RBF et GLF. Nous augmentons la taille de l'environnent de 80 m^2 jusqu'à 200 m^2 dans chaque nouvelle simulation. La **Figure 4.5** montre une exécution sous ARGoS de l'algorithme GLF dans un environnement de taille 140 m².

Taille de l'environnement / Algorithmes	*80*	*100*	*140*	*200*
RBF	870.4	826.4	749	654
GLF	803	666.2	457.4	375.4

Tableau 4.3: Influence de la taille de l'environnement sur les performances.

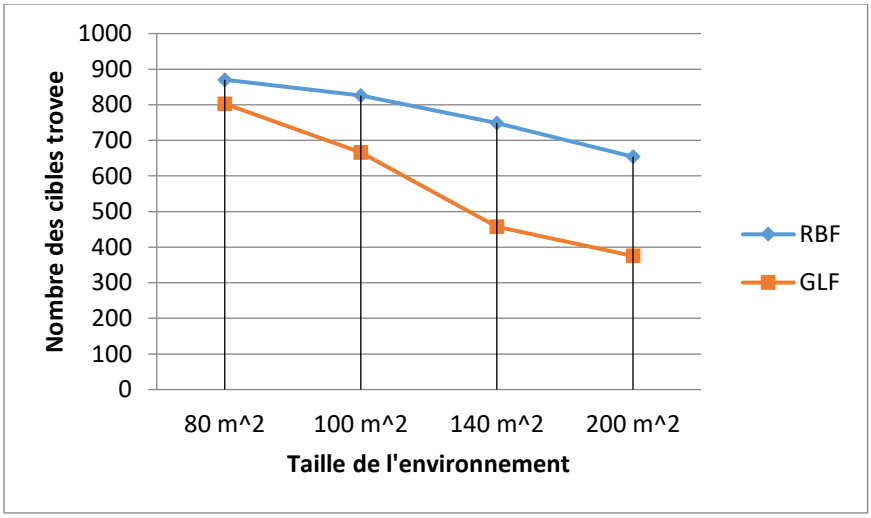


Figure 4.4 : Influence de la taille de l'environnement sur les performances.

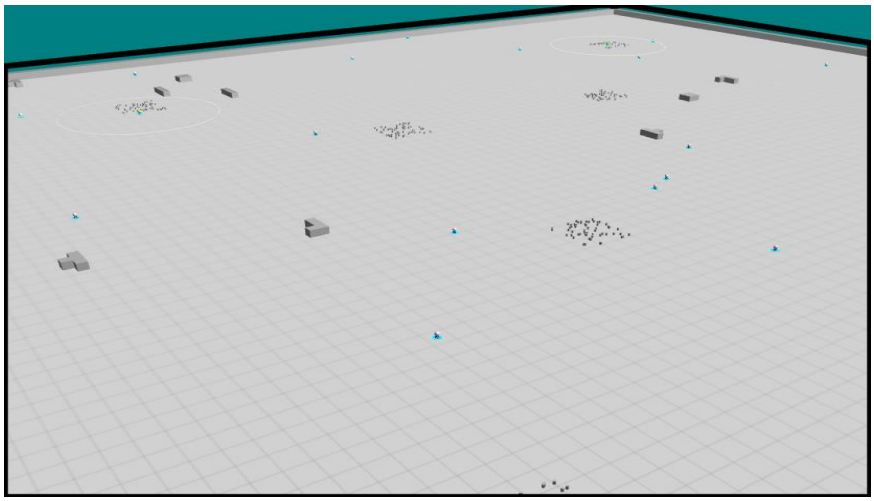


Figure 4.5 : Une exécution sous ARGoS de l'algorithme GLF avec taille de l'environnement 140 m²

Discussion et comparaison

Le **Tableau 4.3** montre les résultats obtenus avec les algorithmes RBF et GLF en augmentant la taille de l'environnement de 80 m² à 200 m². Les résultats sont représentés graphiquement sur la **figure 4.4**. Dans les deux algorithmes, à chaque augmentation de la taille de l'environnement, le nombre de cibles dans l'environnement diminue, Où le nombre de cibles dans la taille de l'environnement de 80 m^2 est de 95,64% et à chaque augmentation, le nombre de cibles trouvées diminue, où dans la taille de l'environnement de 200 ² le nombre de cibles trouvées est 71,8% dans le RBF et dans le GLF, le nombre de cibles diminue de 88,24% à 41,25%, et cela est dû au fait que les robots utilisent la plupart de leur

temps pour explorer plutôt que pour détruire, car la taille de l'environnement augmente, Plus la zone est grande, plus la portée de l'exploration est étendue. Finalement, l'algorithme de RBF donne des bons résultats par rapport à l'algorithme GLF.

3) Scénario 3

Ce scénario montre l'influence du nombre de clusters sur la performance des deux algorithmes RBF et GLF. Nous regroupons les cibles dans des clusters de tailles différentes. Le but est de tester si la distribution des cibles sur plusieurs clusters augmente les performances des algorithmes. Nous varions alors le nombre de clusters de 2 à 16. En dupliquant dans chaque nouvelle simulation le nombre de cluster précédente. La **Figure 4.7** montre une exécution sous ARGoS avec 8 et 2 clusters de l'algorithme RBF.

Nombre de clusters / Algorithmes	*2*	*4*	*8*	*16*
RBF	650	683	701.8	861.2
GLF	508.2	526.8	527.8	675.4

Tableau 4.4: Influence du nombre de clusters sur les performances.

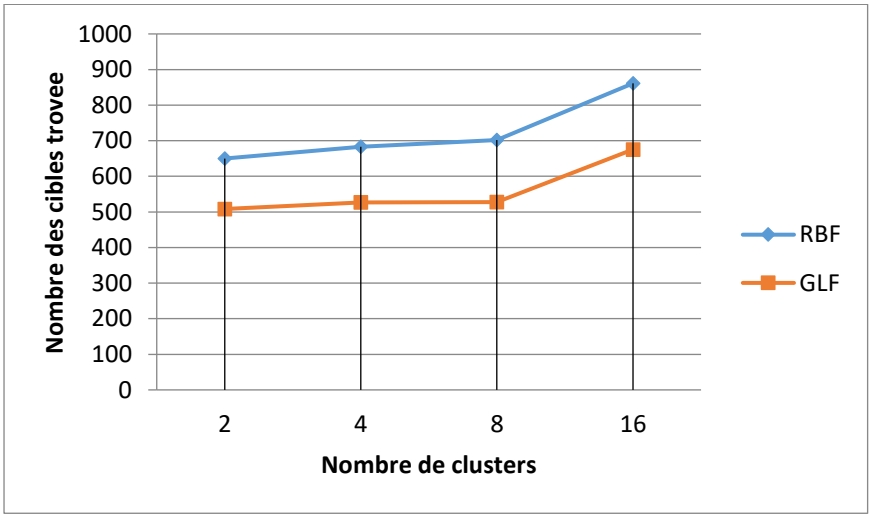
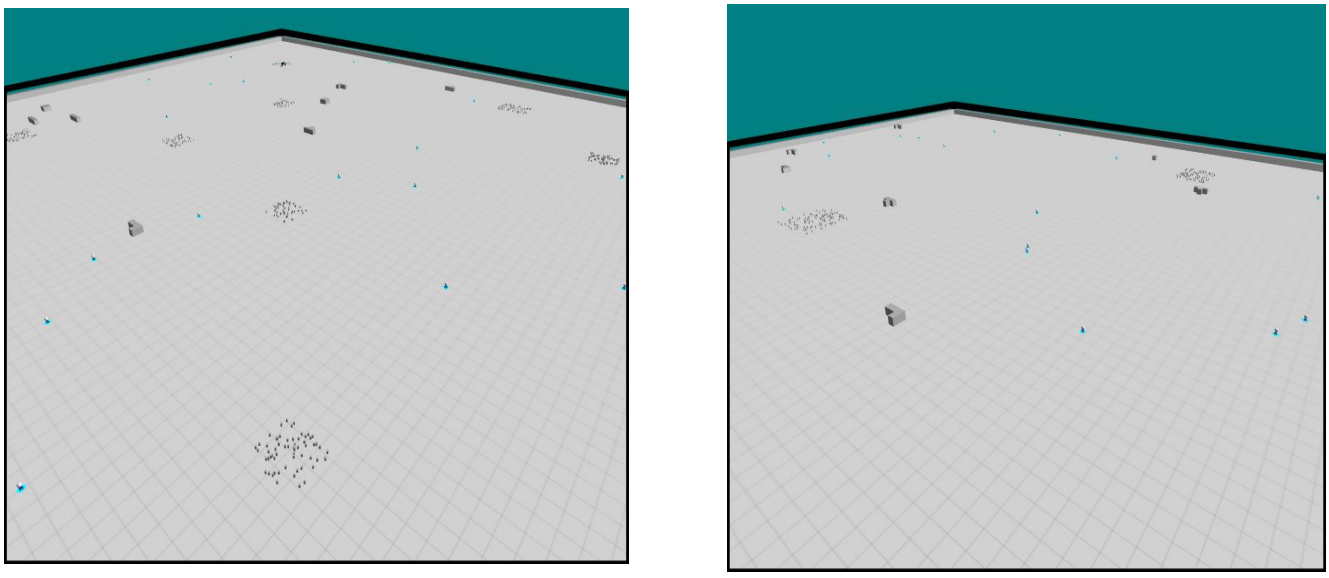


Figure 4.6 : Influence du nombre de clusters sur les performances.



A **B**

Figure 4.7 : Une exécution sous ARGoS de l'algorithme RBF avec 8 cluster (A) et 2 cluster (B)

Discussion et comparaison

Le **Tableau 4.4** et la **Figure 4.6** montrent les résultats obtenus avec les algorithmes RBF, et GLF lorsque on varie le nombre des clusters de 2 à 16. Le nombre de cibles trouvées augmente avec toute augmentation du nombre des clusters. Nous avons remarqué que pour 2 cluster l'algorithme RBF a trouvé 65% des cibles et l'algorithme GLF a trouvé 50.82%, a chaque augmentation de nombre de cluster la performance des algorithmes va augmenter car les robots explorent la majorité des zones dans l'environnement. Avec 16 cluster on a observé que RBF a trouvé 86.12% des cibles et GLF a trouvé 67.54%. On conclut que l'algorithme de RBF donne des bons résultats par rapport à l'algorithme GLF.

4) Scénario 4

Ce scénario montre l'influence du type de distribution des cibles sur la performance des deux algorithmes RBA et GLF. Nous regroupons dans le premier cas les cibles dans des clusters et dans le deuxième cas les cibles sont distribuées aléatoirement de manière Uniforme dans l'environnement. La **Figure 4.9** montre une exécution sous ARGoS de l'algorithme GLF Avec une distribution en clusters (A) et uniforme (B).

Distribution de cibles / Algorithmes	Clusters	Non-Clusters
RBF	562	275.6
GLF	463.4	275.6

Tableau 4.5: Influence de la distribution des cibles sur les performances.

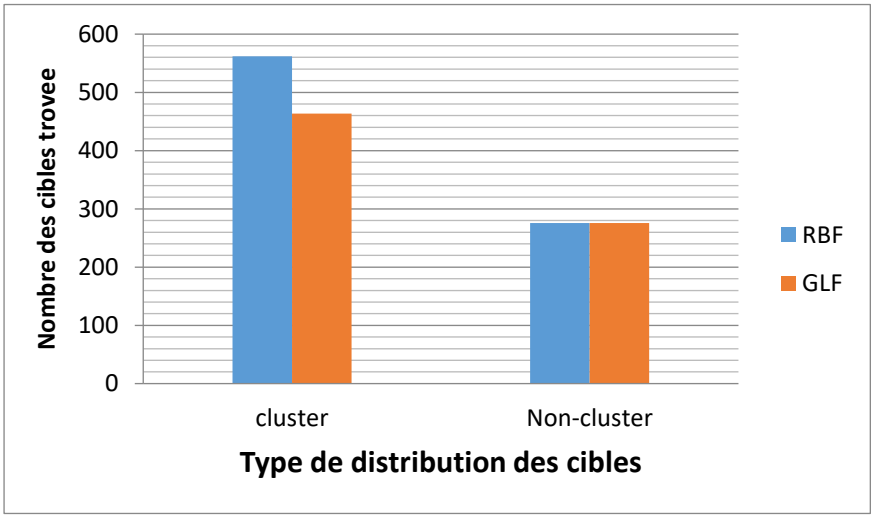
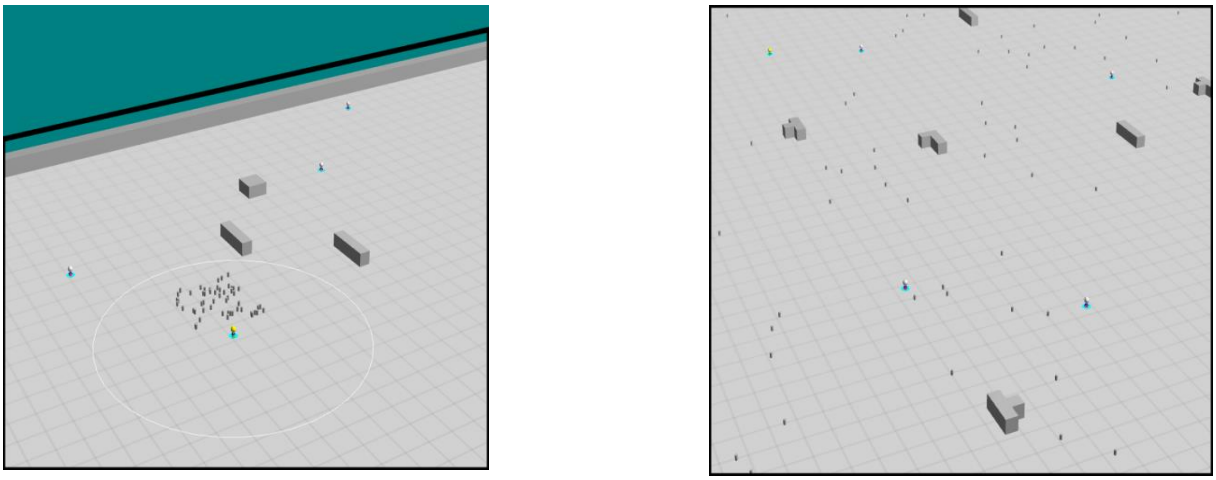


Figure 4.8 : influence de la distribution des cibles sur les performances.



A **B**

Figure 4.9 : Une exécution sous ARGoS illustrant la distribution des cibles : cluster (A), aléatoire (B)

Discussion et comparaison

Le **Tableau 4.5** et la **Figure 4.8** montrent les résultats obtenus. Les performances de deux algorithmes augmentent lorsque les cibles sont regroupées dans des clusters. Où on a observé que RBF a trouvé

93.66% des cibles et GLF a trouvé 77.23%. Les deux algorithmes donnent des bons résultats, mais RBF est toujours mieux que GLF. Les algorithmes RBF et GLF donnent des résultats égaux et mauvais lorsque les cibles sont distribuées aléatoirement (n'atteint même pas 50%), ce qui indique que le caractère aléatoire des stratégies RBF et GLF a un impact négatif sur l'exploration globale.

Les résultats obtenus montrent que RBF est robuste aux variations de l'environnement par rapport au GLF lorsque le type de distribution cible est sous forme des clusters, tandis que les deux sont très mauvais lorsque les cibles sont distribuées aléatoirement.

V Conclusion

Dans ce chapitre nous avons présenté la plateforme ARGoS utilisé pour l'implémentation de nos algorithmes. Nous avons présenté aussi les robots Foot-bot, ainsi que leurs caractéristiques. Ensuite, Nous avons expliqué le comportement des algorithmes améliorés RBF et GLF. Nous avons ainsi analysé les performances des deux algorithmes RBF et GLF. De ce fait, nous avons proposé un ensemble de scénarios avec variation de certains paramètres pour voir s'ils ont une influence sur les performances. Dans toutes les simulations, les deux stratégies donnent de bons résultats, mais BRF montre toujours sa supériorité sur GLF. Cela prouve quantitativement que la stratégie d'exploration BRF est plus efficace que la GLF.

Conclusion générale et perspectives

L'utilisation des algorithmes issus de l'intelligence en essaim pour la résolution collective de problèmes et devenue très répondue en robotique mobile. Parmi ces problèmes : le déminage, le nettoyage, le sauvetage…etc. L'exploration de l'environnement constitue un pilier dans ces différentes applications. Une exploration Multi-Robots consiste à naviguer dans l'espace selon une stratégie de déploiement dans le but de chercher des objets précieux. La marche varie d'une marche aléatoire, stratégique ou hybride. Les travaux relies montrent que la marche aléatoire donne plus de chance à localiser des objets dans des environnements inconnus.

Dans ce travail nous avons concentré sur la dispersion des robots par division implicite de l'environnement à travers la lumière. Ce comportement de répulsion est inspiré du comportement d'attraction des lucioles (firefly). Nous avons proposé deux algorithmes RBF et GLF qui améliorent les algorithmes RB et GL respectivement par le comportement répulsive en utilisant la lumière. Une telle exploration a permis de localiser plus des cibles dans l'environnement.

Les algorithmes ont été implémentés sous ARGoS et les résultats de simulations montrent que le RBF est plus performant que le GLF.

Comme perspectives à ce travail, nous souhaitons :

1. Etendre le comportement des robots pour prendre en charge la limitation d'énergie des robots ;
2. Introduire la lumière dans l'exploration en résolvant le problème qui apparut dans cette conception ;
3. Etudier les paramètres Lévy pour générer des pas plus large et assurer un comportement plus efficace du GLF.

Bibliographie

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

[Beni et Wang, 1989] Beni, G., & Wang, J. (1989). Swarm Intelligence in Cellular Robotic Systems, Proceed. NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, June 26-30. Y.: NATO.

[Beni, 2004] Beni, G. (2004, July). From swarm intelligence to swarm robotics. In International Workshop on Swarm Robotics (pp. 1-9). Springer, Berlin, Heidelberg.

[Beni, 2005] Beni, G. (2005). From swarm intelligence to swarm robotics. vol. 3342 of Lecture Notes in Computer Science, pp. 1–9.

[Blum et Merkle, 2008] Blum, C. et Merkle, D., éditeurs (2008). Swarm Intelligence. Natural Computing Series. Springer-Verlag.

[Benavides et al.,2016] Benavides, F., Monzón, P., Chanel, C. P. C., & Grampín, E. (2016, October). Multi-robot Cooperative Systems for Exploration: Advances in dealing with constrained communication environments. In 2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR) (pp. 181-186). IEEE.

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

[Charrier et al.,2007] Charrier, R., Bourjot, C., & Charpillet, F. (2007, November). Un modèle connexionniste pour l'intelligence en essaim : le système multi-agent logistique. In Colloque de l'Association pour la Recherche Cognitive-ARCo'07 : Cognition–Complexité–Collectif (pp. 19-32).

[Charrier, 2009] Charrier Rodolphe. L'intelligence en essaim sous l'angle des systèmes complexes : étude d'un système multi-agent réactif à base d'itérations logistiques couplées. Modélisation et simulation. Université Nancy II, 2009. Français.

[Dorigo et Sahin, 2004] Dorigo, M., & Sahin, E. (2004). Guest editorial: Swarm robotics. Autonomous Robotics, 17(2-3), 111-113.

[De Rango et al.,2018] De Rango, F., Palmieri, N., Yang, X. S., & Marano, S. (2018). Swarm robotics in wireless distributed protocol design for coordinating robots involved in cooperative tasks. Soft Computing, 22(13), 4251-4266.

[Dimidov et al.,2016] Dimidov, C., Oriolo, G., & Trianni, V. (2016, September). Random walks in swarm robotics: an experiment with kilobots. In International Conference on Swarm Intelligence (pp. 185-196). Springer, Cham.

[El-Shorbagy et Hassanien., 20018] El-Shorbagy, M. A., & Hassanien, A. E. (2018). Particle Swarm Optimization from Theory to Applications. International Journal of Rough Sets and Data Analysis, 5(2), 1–24.

[Esmin et al.,2015] Esmin, A. A., Coelho, R. A., & Matwin, S. (2015). A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. *Artificial Intelligence Review*, *44*(1), 23-45.

[Fricke et al.,2016] Fricke, G. M., Hecker, J. P., Griego, A. D., Tran, L. T., & Moses, M. E. (2016, October). A distributed deterministic spiral search algorithm for swarms. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 4430-4436). IEEE.

[Gasparri et al., 2012] Gasparri, A., Oriolo, G., Priolo, A., & Ulivi, G. (2012, October). A swarm aggregation algorithm based on local interaction for multi-robot systems with actuator saturations. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 539-544). IEEE.

[Garnier, 2008] Garnier, S. (2008). Décisions collectives dans des systèmes d'intelligence en essaim (Doctoral dissertation, Université de Toulouse, Université Toulouse III-Paul Sabatier).

[Hamann,2018] Hamann, H. (2018). Swarm robotics: A formal approach (p. 9). Springer International Publishing.

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

[Idiri, 2018] Idiri, M. (2018). Une stratégie d'exploration pour des applications de la robotique mobile.

[Isaacs et al., 2020] Isaacs, J. T., Dolan-Stern, N., Getzinger, M., Warner, E., Venegas, A., & Sanchez, A. (2020, April). Central Place Foraging: Delivery Lanes, Recruitment and Site Fidelity. In *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)* (pp. 319-324). IEEE.

[Ijspeert et al., 2001] Ijspeert, A. J., Martinoli, A., Billard, A., & Gambardella, L. M. (2001). Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment. Autonomous Robots, 11(2), 149-171.

[Jevtić et Andina, 07] Jevtić, A., & Andina de la Fuente, D. (2007). Swarm intelligence and its applications in swarm robotics.

[Kennedy et al., 2001] Kennedy, J., Eberhart, R.C. and Shi, Y. (2001) Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco.

[Karaboga., 2005] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200, pp. 1-10). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

[Kennedy et Eberhart., 1995] Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks (Vol. 4, pp. 1942–1948). IEEE

[Kube & Bonabeau, 2000] Kube, C. R., & Bonabeau, E. (2000). Cooperative transport by ants and robots. Robotics and autonomous systems, 30(1-2), 85-101.

[Kaminka et al, 08] Kaminka, G. A., Schechter-Glick, R., & Sadov, V. (2008). Using sensor morphology for multirobot formations. IEEE Transactions on Robotics, vol. 24(2), pp. 271-282.

[Kaldé et al, 14] Kaldé, N., Charpillet, F., & Simonin, O. (2014, October). Comparaison de stratégies d'exploration multi-robot classiques et interactives en environnement peuplé. In Vingt-deuxièmes journées francophones sur les systèmes multi-agents, Loriol-sur-Drôme, France, Octobre 8-10, 2014. Cépaduès.

[Katada et al, 16] Katada, Y., Nishiguchi, A., Moriwaki, K., & Watakabe, R. (2016). Swarm robotic network using Lévy flight in target detection problem. Artificial Life and Robotics, 21(3), 295-301.

[Khaluf et al, 2018] Khaluf, Y., Van Havermaet, S., & Simoens, P. (2018, September). Collective Lévy walk for Efficient Exploration in Unknown Environments. In International Conference on Artificial Intelligence : Methodology, Systems, and Applications (pp. 260-264). Springer, Cham.

[Le et Plaku., 2018] Le, D., & Plaku, E. (2018, July). Multi-Robot Motion Planning with Dynamics Guided by Multi-Agent Search. In *IJCAI* (pp. 5314-5318).

[Li et al., 2019] Li, S., Li, W., Zhang, H., & Wang, Z. (2019). Research and implementation of parallel artificial bee colony algorithm based on ternary optical computer. Automatika, 60(4), 422-431.

[Li et al.,2016] Li, W., Gauci, M., & Groß, R. (2016). Turing learning: a metric-free approach to inferring behavior and its application to swarms. Swarm Intelligence, 10(3), 211-243.

[Martinoli et al., 1999] Martinoli, A., Ijspeert, A. J., & Mondada, F. (1999). Understanding collective aggregation mechanisms: From probabilistic modelling to experiments with real robots. Robotics and Autonomous Systems, *29*(1), 51-63.

[Nicole et al., 2014] Nicole El Zoghby, Valeria Loscri, Enrico Natalizio, Véronique Cherfaoui. Robot Cooperation andSwarm Intelligence. Wireless Sensor and Robot Networks : From Topology Control to CommunicationAspects, World Scientific Publishing Company, pp.168-201, 2014.

[Pang et al., 2019] Pang, B., Song, Y., Zhang, C., Wang, H., & Yang, R. (2019). A swarm robotic exploration strategy based on an improved random walk method. Journal of Robotics, 2019.

[Palmieri et al., 2018] Palmieri, N., Yang, X. S., De Rango, F., & Santamaria, A. F. (2018). Self-adaptive decision-making mechanisms to balance the execution of multiple tasks for a multi-robots team. Neurocomputing, 306, 17-36.

[Palmieri et al., 2019] Palmieri, N., Yang, X. S., De Rango, F., & Marano, S. (2019). Comparison of bio-inspired algorithms applied to the coordination of mobile robots considering the energy consumption. Neural Computing and Applications, 31(1), 263-286.

[Palmieri et al., 2015] Palmieri, N., De Rango, F., Yang, X. S., & Marano, S. (2015, November). Bio-inspired Strategies for the Coordination of a Swarm of Robots in an Unknown Area. In International Joint Conference on Computational Intelligence (pp. 96-112). Springer, Cham.

[Palmieri et al., 2016] Palmieri, N., & Marano, S. (2016). Discrete firefly algorithm for recruiting task in a swarm of robots. In Nature-Inspired Computation in Engineering (pp. 133-150). Springer, Cham.

[Palmieri et al., 2015] Palmieri, N., De Rango, F., Yang, X. S., & Marano, S. (2015, November). Multi-robot cooperative tasks using combined nature-inspired techniques. In 2015 7th International Joint Conference on Computational Intelligence (IJCCI) (Vol. 1, pp. 74-82). IEEE.

[Sutantyo et al., 2013] Sutantyo, D., Levi, P., Möslinger, C., & Read, M. (2013, August). Collective-adaptive lévy flight for underwater multi-robot exploration. In 2013 IEEE International Conference on Mechatronics and Automation (pp. 456-462). IEEE.

[Suárez et Iglesias., 2017] Suárez, P., & Iglesias, A. (2017, February). Bat algorithm for coordinated exploration in swarm robotics. In International Conference on Harmony Search Algorithm (pp. 134-144). Springer, Singapore.

[Sahin et al., 2008] Sahin, E., Girgin, S., Bayindir, L. et Turgut, A. E. (2008). Swarm

robotics. In Blum, C. et Merkle, D., _editeurs: Swarm Intelligence, Natural Computing Series, page 87{100. Springer-Verlag.

[Santos et al, 2018] Santos, R. G., de Freitas, E. P., Cheng, S., de Almeida Ribeiro, P. R., & de Oliveira, A. C. M. (2018, November). Autonomous Exploration Guided by Optimisation Metaheuristic. In 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) (pp. 1759-1764). IEEE.

[Sahin, 04] Şahin, E. (2004). Swarm robotics: From sources of inspiration to domains of application. In International workshop on swarm robotics (pp. 10-20). Springer, Berlin, Heidelberg.

[Soysal et Şahin., 2006] Soysal, O., & Şahin, E. (2006, September). A macroscopic model for self-organized aggregation in swarm robotic systems. In International Workshop on Swarm Robotics (pp. 27-42). Springer, Berlin, Heidelberg.

[Sànchez et al., 2018] Sànchez, N. D. G., Vargas, P. A., & Couceiro, M. S. (2018, July). A Darwinian Swarm Robotics Strategy Applied to Underwater Exploration. In 2018 IEEE Congress on Evolutionary Computation (CEC) (pp. 1-6). IEEE.

[Santos et al., 2018] Santos, R. G., de Freitas, E. P., Cheng, S., de Almeida Ribeiro, P. R., & de Oliveira, A. C. M. (2018, November). Autonomous Exploration Guided by Optimisation Metaheuristic. In 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) (pp. 1759-1764). IEEE.

[Sharma et al., 2012] Sharma, T. K., Pant, M., & Singh, V. P. (2012). Improved local search in artificial bee colony using golden section search. arXiv preprint arXiv:1210.6128.

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

[Tran et al, 2004] Tran, T., Nguyen, T. T., & Nguyen, H. L. (2014). Global optimization using Levy flights. arXiv preprint arXiv:1407.5739.

[Yang, 2009] Yang, X. S. (2009, October). Firefly algorithms for multimodal optimization. In International symposium on stochastic algorithms (pp. 169-178). Springer, Berlin, Heidelberg.

[Yang, 2010] Yang, X. S. (2010). Nature-inspired metaheuristic algorithms. Luniver press.

[Zoghby et al., 2014] Zoghby, N. E., Loscri, V., Natalizio, E., & Cherfaoui, V. (2014). Chapter 8 : Robot Cooperation and Swarm Intelligence. In Wireless Sensor and Robot Networks : From Topology Control to Communication Aspects (pp. 163-201).

Webographie

[1] Solved: the mystery of why locusts swarm, URL : https://www.independent.co.uk/news/science/solved-the-mystery-of-why-locusts-swarm-1520409.html . Dernière consultation : 12/9/2020.

[2] Universe Bats, une étude pour sauver les amis des éleveurs, URL : https://napoli.repubblica.it/cronaca/2017/04/12/news/universo_pipistrelli_uno_studio_per_salvare_gli_amici_degli_allevatori-162791609/ .Dernière consultation : 12/9/2020.

[3] Pingouin ou manchot, Quelle différence ?, URL : https://www.animaux-online.com/ARTICLE,LECTURE,1829_PINGOUIN-OU-MANCHOT-QUELLE-DIFFERENCE-.HTML .Dernière consultation: 12/9/2020.

[4] Large Grey Wolf Pack (15) run the gauntlet!, URL : https://comicvine.gamespot.com/forums/battles-7/large-grey-wolf-pack-15-run-the-gauntlet-1942907/?page=1 . Dernière consultation : 12/9/2020.

[5] Vie artificielle : les systèmes inspirés de la nature, URL : https://www.futura-sciences.com/tech/dossiers/robotique-vie-artificielle-systemes-inspires-nature-262/page/7/ .Dernière consultation : 12/9/2020.

[6] Somalie invasion criquet, URL : **https://dailygeekshow.com/somalie-invasion-criquet/** .Dernière consultation : 12/9/2020.

[7] L'intelligence collective, cette étonnante capacité du vivant, URL : https://www.4emesinge.com/lintelligence-collective-cette-etonnante-capacite-du-vivant/ . Dernière consultation : 12/9/2020.

[8] BRICS // Can SWARM Intelligence help Online Luxury Market? URL https://medium.com/@paolofranzesejun/brics-can-swarm-intelligence-help-online-luxury-market-cc5d72d8c6e1 . Dernière consultation : 12/9/2020.

[9] Essaimage d'abeilles : focus sur la duplication de la colonie, URL : https://www.trigobert.net/essaimage-dabeilles-duplication-de-colonie.html. Dernière consultation : 12/09/2020.

[10] Ant Colony Optimization, URL : http://praneetsamaiya.blogspot.com/2012/02/ant-colony-optimization.html . Dernière consultation : 12/9/2020.

[11] foot-bot, URL : https://www.pinterest.com/pin/162129655311149116/ . Dernière consultation : 12/9/2020.

[12] E-puck, URL : https://perfectocean.wordpress.com/2012/01/24/hello-world/. Dernière consultation : 12/9/2020.

[13] 3 promising UK-based drone startups that are raising money today, URL: https://medium.com/@jadjamous/3-promising-european-drone-startups-that-are-raising-84682e0608df . Dernière consultation : 12/9/2020.

[14] Blood stream nano bots, URL : https://www.pinterest.com/pin/425238389793811328/. Dernière consultation : 26/5/2018.

[15] Swarming Robots, URL : https://www.pinterest.co.uk/pin/22377329371503943/. Dernière consultation : 12/9/2020.

[16] Swarm robots reduce human error, URL https://translate.google.com/#view=home&op=translate&sl=auto&tl=en&text=Swarm%20robots%20reduce%20human%20error. Dernière consultation : 12/9/2020.

[17] Cyborg, URL : https://sdtimes.com/buzz/researchers-develop-buzz-a-programming-language-for-robot-swarms/.Dernière consultation: 12/9/2020.

[18] Three s-bots passing a gap in swarm-bot configuration, URL : https://www.researchgate.net/figure/Three-s-bots-passing-a-gap-in-swarm-bot-configuration_fig3_221531250.Dernière consultation: 12/9/2020.

[19] RoboCup Soccer, URL : http://www.robocup2014.org/?p=893.Dernière consultation: 12/9/2020.

[20] ARGoS Large-scale robot simulations, URL: https://www.argos-sim.info/. Dernière consultation : 12/9/2020.

[21] Simulateur Player Stage, URL : http://playerstage.sourceforge.net/index.php?src=stage. Dernière consultation : 12/9/2020.

[22] Cyberbotics, URL : https://www.cyberbotics.com/#cyberbotics.Dernière consultation: 12/9/2020.

[23] Gazebo, URL : https://doc.ubuntu-fr.org/gazebo.Dernière consultation: 12/9/2020.