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 Fin d' Étude Master

Filière : Informatique

Option : Sciences et Technologies de l'Information et de la communication

Thème :

Réalisation d'un système de suivi d'objets basé sur les Drones

Par
Abderrezzag ZIOU

Devant le jury

Dr. Said BRAHIMI	Université de Guelma	Président
Dr. Brahim FAROU	Université de Guelma	Encadreur
Dr. Karima BENHAMZA	Université de Guelma	Examinateur

Octobre 2020

Remerciements

J'offre premièrement de sincères et chaleureux remerciements au Dieu qui m'a aidé à terminer ce travail, mon encadreur Dr. Farou Brahim et son doctorant M.Benrazk Alaedine. Le mérite d'un mémoire appartient certes à l'auteur, mais également à son directeur qui l'encadre. Dans mon cas, mes directeurs ont été d'un soutien et d'une attention exceptionnels. La confiance qu'ils m'ont accordée ainsi que le soutien moral qu'ils ont manifesté à mon égard m'ont permis d'accumuler des expériences professionnelles et personnelles marquantes qui font de moi une personne grandie. Je salue leur amabilité, leur patience, leur disponibilité, leur souplesse d'esprit et leur savoir-faire. C'est certes avec joie et fierté que je dépose aujourd'hui ce mémoire, mais aussi avec un brin de nostalgie que je termine mon programme d'études . Je tiens également à exprimer ma gratitude à ma famille, mon père décédé qui a fait de moi un homme, ma mère, mes frères et mes soeurs qu'ils me donnent un coup de main a chaque fois que je tombe. Je veux remercie mes amis et chaque personne qui m'a donné de l'aide dans tout moment de détresse.

Merci beaucoup.

Dédicaces

Je dédie ce modeste travail à : A mes parents. Aucun hommage ne pourrait être à la hauteur de l'amour Dont ils ne cessent de me combler. Que dieu leur procure bonne santé et longue vie. A celui que j'aime beaucoup et qui m'a soutenue tout au long de ce projet : ma mère et ma grandes seour, et bien sur A mes frères MIDOU, AYMEN, SALAH, YAHIYA et aussi mon frères ZINEDIN BELGHARBI, sans oublié. A toute ma famille, et mes amis, ZAIDOU, SAJED, MOUHAMED, JOURI, MERIEM et toute la famille. Et à tous ceux qui ont contribué de près ou de loin pour que ce projet soit possible, je vous dis merci. Je vous aime.

RÉSUMÉ

Le suivi des objets mobile par un seul drone ne permet pas de fournir une performance maximale à cause de leur limitation dans la capacité de traitement et l'autonomie nécessaire pour l'accomplissement de la mission. Dans ce mémoire, nous allons proposer un système de surveillance qui permet de gérer et de coordonner un essaim de drones dans la tache de surveillance. Notre approche opère sur trois principales phases. La première consiste à effectuer une recherche globale dans la zone à surveiller via l'algorithme "Levy Flights". Dès la détection d'un problème, le drone se bascule vers une recherche locale via l'algorithme "Fireflies". Ce dernier permet au drone de se déplacer localement d'un problème à un autre afin de localiser le plus important. Lorsque le système se stabilise, il commence l'exécution de l'algorithme "Grey Wolves" pour maximiser le champ de vision vu qu'un seul drone n'est pas suffisant. Pour garantir une efficacité maximale, une stratégie de remplacement, en cas de panne de batterie, est proposée afin de garantir la continuité de la mission. Les algorithmes proposés en été implémenté et tester par notre propre simulateur. Les résultats obtenus sont encourageants.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GÉNÉRALE

Dans les dernières années, il y a eu un intérêt considérable pour le développement et l'amélioration des outils de surveillance soit dans les endroits publics ou privés. Les véhicules aériens sans pilote (UAV) ou drone sont parmi les nouvelles technologies qui sont utilisées dans le contexte de la vidéosurveillance. Les drones sont des objets volants sans pilote à bord, ils sont équipés par divers capteurs (tels que des caméras) et ils possèdent une capacité de traitement qui permet un déploiement facile, une grande mobilité et un large champ de vision.

Dans le contexte du suivi des objets mobile, un seul drone n'est pas suffisant pour fournir des taches opérationnelles complètes, en raison de leur limitation dans : la capacité de traitement, l'efficacité et la qualité de la mission, la flexibilité dans l'exécution des tâches, le temps de vol (énergie). Pour surmonter les limitations des drones dans les systèmes de surveillance, les récents travaux scientifiques ont mis l'accent sur la modélisation des essaims pour mieux comprendre comment des animaux sociaux interagissent et atteignent leurs objectifs (Swarm UAV). Ces derniers sont utilisés dans de nombreuses applications telles que l'exploration, la surveillance, les attaques et la cartographie d'environnements inconnus. Techniquement, un «UAV essaim»

est un groupe d'UAV communiquant les uns avec les autres de manière autonome à l'aide d'un algorithme de contrôle intelligent embarqué. Dans le but de réduire le coût du système de surveillance en termes d'énergie, bande passante et capacité stockage et d'améliorer la qualité du suivi des objets mobiles, nous allons proposer une nouvelle approche distribuée basée sur les algorithmes méta heuristiques qui permet de gérer et coordonner des drones pour le suivi des objets en mouvement.

Notre système a été divisé en trois principales phases ; chacune d'elle est gérée par un algorithme méthaeuristiques. La première phase appelée phase de recherche permet en premier temps de surveiller la zone pour détecter d'éventuel problème en utilisant l'algorithme de recherche globale "Levy Flights". Dès la réception d'un signal, le système se bascule vers une recherche locale guidée par l'algorithme "Fireflies". La deuxième phase permet de coordonner la communication entre les drones en utilisant l'algorithme du "Grey Wolves". La dernière phase consiste à remplacer les drones avec des batteries faibles par d'autres drones ayant plus d'énergie. Ce mémoire est organisé en quatre chapitres, précédé par une introduction générale et terminé par une conclusion et quelques pertinentes perspectives.

Le premier chapitre est dédié aux drones. Il présente en détail les composants d'un drone, ses caractéristiques techniques et son domaine d'application. La suite du chapitre aborde les systèmes multi drones en termes de coopération et de distribution des tâches ainsi que les différentes architectures de communication. Le chapitre se termine par un bref état de l'art sur les systèmes de surveillance basés sur une plate-forme multi drones.

Le deuxième chapitre est un état de l'art sur les métaheuristiques bio-inspirés. Il commence par les concepts et définitions liés à la métaheuristique bio-inspirée et à l'intelligence des essaims. Ensuite, un tour d'horizon sur les algorithmes métaheuristiques bio-inspirés proposés dans la littérature est

présenté. Le reste du chapitre contient une description détaillée des méta-heuristiques bio-inspirés utilisées dans ce mémoire.

Dans le troisième chapitre, le manuscrit introduit l'architecture principale du système proposé et présente en détail les différents modules qui le composent.

Le dernier chapitre commence par définir les langages de programmation et les plates-formes utilisés pour le développement. La suite du chapitre est consacrée à la présentation du simulateur qui a été développé sur la base des algorithmes proposés précédemment. Ce chapitre se termine par un petit aperçu sur les résultats de la simulation effectuée en utilisant des données aléatoires.

CHAPITRE 1
LES DRONES DE SURVEILLANCE

1.1 Introduction

Véhicules aériens télépilotés (RPV [1]), aéronef télépiloté (RPA [2]), aéronef télécommandé (ROA [3]), véhicules aériens sans pilote (UAV [4]) ou systèmes d'aéronef sans pilote (UAS [5]) sont des acronymes de noms de drones qui désignent des avions sans pilote humain à bord. Les drones sont des technologies émergentes qui, au cours de la dernière décennie, ont atteint un potentiel considérable dans les domaines militaire (exploration aérienne, surveillance du champ de bataille, localisation des cibles, suivi, évaluation des dommages et arrestations antiterroristes), public et civil (surveillance, transport, surveillance environnementale, surveillance industrielle, services agricoles et secours en cas de catastrophe). Cela est en raison de leur faisabilité économique, de leur facilité d'utilisation et de la possibilité de se déplacer

1. RPV : Remotely Piloted Vehicle
2. RPA : Remotely Piloted Aircraft
3. ROA : Remotely Operated Aircraft
4. UAV : Unmanned Aerial Vehicles
5. UAS : Unmanned Aircraft Systems

dans des endroits que les humains ne peuvent pas atteindre.

Dans une mission de surveillance (suivi, localisation des cibles , évaluation des dommages en cas de catastrophe, etc.), les drones peuvent être équipés de différents types d'équipements de surveillance, tels que des caméras, capteur thermique, etc. ; qui sont utilisés pour collecter des informations sur l'environnement afin d'assurer une surveillance sophistiquée. Cependant, la complexité de la mission de surveillance et l'environnement surveillé sont des obstacles majeurs aux systèmes basées sur un seul drone. En effet, un seul drone ne peut assurer que des missions opérationnelles limitées. Dans ce contexte, les systèmes multi-drone et en particulier les systèmes dits essaims de drones, nous permettent de palier une partie du problème. La collaboration et la synchronisation efficaces de plusieurs drones permettent de construire un système qui utilise efficacement les drones par rapport à un seul drone.

1.2 Véhicules aériens sans pilote : Un bref aperçu

1.2.1 Définition

Le premier aéronef sans pilote qui a touché le ciel remonte à 1917 par le français Max Boucher [25]. Depuis cette époque, diverses définitions ont été proposées dans la littérature. Dans cette sous-section, nous présentons quelques définitions plus récentes.

Définition 1 *"Le nom de drone vient d'un mot anglais signifiant faux-bourdon donné comme surnom par l'artillerie anglaise dans les années 1930 à un avion cible utilisé pour l'entraînement ayant un vol lent et bruyant ressemblent à celui du bourdon. Maintenant il désigne un aéronef sans pilote à bord généralement télécommandé*

du sol, il peut être programmé pour voler de façon autonome ou via un smart-phone ou une tablette, etc." [62].

Définition 2 *"Un drone ou Unmanned Aerial Vehicle (UAV) est un aéronef sans passager ni pilote qui peut voler de façon autonome ou être contrôlé à distance depuis le sol. Le mot « drone » est une extrapolation d'un terme anglais qui signifie « faux-bourdon ». En français, le terme est employé pour désigner des véhicules aériens, terrestres, de surface ou sous-marins, alors que la classification anglo-saxonne distingue chaque type d'appareil" [71].*

Définition 3 *"Les drones sont des engins volants de taille réduite, moins chers et plus simples à mettre en œuvre qu'un avion, ils sont plus discrets et leur perte est moins grave que celle d'un appareil de son pilote. Leur taille varie de quelques centimètres à plusieurs dizaines de mètres. Leurs formes également, tout comme leurs types de propulsion : certains sont équipés de réacteurs, d'autres d'hélices, quand d'autres utilisent des rotors, à l'instar des hélicoptères" [12].*

Définition 4 *"Un véhicule aérien sans pilote (UAV) (ou véhicule aérien sans équipage, communément appelé drone) est un aéronef sans pilote humain à bord et un type de véhicule sans pilote. Les UAV font partie d'un système d'aéronef sans pilote (UAS), qui comprend un UAV, un contrôleur au sol et un système de communication entre les deux. Le vol des drones peut fonctionner avec différents degrés d'autonomie : soit sous contrôle à distance par un opérateur humain, de manière autonome par des ordinateurs de bord ou piloté par un robot autonome" [86].*

1.2.2 Architecture et Composants

Pour simplifier au maximum, un drone est composé de trois parties principales [80] :

- ✓ Le châssis (ou frame), c'est un peu comme le squelette du drone. Il peut prendre différentes formes selon les modèles et selon son nombre de bras. Ainsi, il existe des drones tricoptères, quadricoptères, hexacoptères, etc. En aluminium, en plastique, en fibre de carbone ou même en bois, le châssis peut véritablement varier selon les modèles de drones.

- ✓ Le système de propulsion est quant à lui composé de moteurs que l'on appelle plus précisément rotors, d'hélices, de contrôleurs de vitesse électriques (ESC) et d'une batterie lipo (lithium polymère).

- ✓ Le contrôleur de vol sert à établir le lien entre le drone et le pilote, via un récepteur connecté. C'est un circuit intégré doté d'un microprocesseur, de capteurs, et de broches d'entrée et de sorties.

En plus des trois principaux composants mentionnés ci-dessus, les drones peuvent contenir d'autres composants complémentaires selon la mission qui leur est assignée :

- ✓ Une caméra infrarouge, détectant toute source de chaleur (humaine, animale, ou provenant d'un véhicule).

- ✓ Une caméra capable de retransmettre en temps réel ce qui se passe sur le terrain.

- ✓ Des gyroscopes pour se stabiliser et contrebalancer leurs mouvements.

1.2.3 Caractéristiques

D'après [2] le drone est plus pratique par rapport aux autres machines (hélicoptère, avion, bateau, etc.), et tout ça par ce qu'il est :

- ✓ **Précis et Maniable :** Permet le maintien d'une position GPS et d'une altitude.

- ✓ **Légal Sécurisé :** Les machines et les pilotes disposent d'équipements de sécurité requis et sont conformes avec la réglementation.
- ✓ **Rapide :** Mise en oeuvre ultra-rapide, moins de 10 minutes pour déployer le système.
- ✓ **Économique :** Moins cher que les solutions traditionnelles (hélicoptère, avion, etc.).
- ✓ **Spectaculaire :** Images hors du commun et angles inédits.
- ✓ **Stable :** Dispositif favorisant la prise de vue de qualité.
- ✓ **Respecte L'environnement :** Silencieux et sans émission de CO2.
- ✓ **Proximité :** Vol au plus près du sujet.
- ✓ **Accès :** ils peuvent accéder facilement à des endroits impossibles pour l'être humain.

1.2.4 Classification

la classification des drone peut être difficile car elle se diffère dans chaque pays . Elle dépend de plusieurs paramètres tels que l'altitude de vol, capacité du batterie, le poids et la taille des drones, l'autonomie de vol, l'endurance, la vitesse, les ailes, etc[11]. Il y a notamment des drones Hale (High Altitude Long Endurance), des drones MALE (Medium Altitude Long Endurance), des drones à courte et moyenne portée, des mini drones et des micro drones (MAV). Ils se distinguent également selon leurs fonctions : drones tactiques, drones stratégiques et drones de combat (Unmanned Combat Air Vehicle UCAV). De plus, le type d'engin peut également les différencier : voilure fixe, voilure tournante et systèmes hybrides[23].

Voici un tableau (1.1)comparatif qui contient les caractéristiques qui ont motionné précédemment [1].

	Mini et Micro UAVs	Tactical UAV	MALE UAVs	HALE UAVs
Altitude	< 300 m	< 5000 m	5000-15000 m	Max 20000 m
Poids	Micro :<500g, Mini :20 kg	100-500 kg	1800 kg	12000 kg
Application	Civil ou Commercial	military	military	military
Autonomie	Micro30 min Minifew hours	10 hours	24 hours	UAV Global Hawk : 35 hours

TABLE 1.1 – Types de drones et leurs caractéristiques

1.2.5 Enjeux

Les drones sont utilisés dans de nombreux domaines en ensemble puis plus particulièrement en flotte[3]. Ils sont choisis grâce à :

Résilience : La pluralité des plateformes constituant un essaim offre la résilience du système dans le cas de la perte de quelques engins. Dans le cas de l'utilisation d'une unique plateforme, la défaillance du véhicule met en péril la mission.

Coût : Le coût d'achat et de maintenance de plusieurs drones d'envergure de l'ordre d'un mètre est significativement inférieur à celui d'un drone tactique, MALE ou HALE.

Discrétion : La détection de plusieurs drones est plus complexe que celle d'un appareil de grande envergure, ce qui peut être crucial dans le cadre d'applications militaires.

Décollage et atterrissage : Les drones n'ont besoin d'aucune infrastructure pour décoller et atterrir. Les voilures fixes, en particulier de grande envergure, nécessitent une grande surface libre pour décoller et atterrir.

Charge de travail pour l'opérateur : La répartition des tâches entre les drones d'un essaim se fait au sein même de l'essaim ; du point de vue de l'opérateur, l'essaim est donc considéré comme une entité unique. La gestion d'une mission n'est donc pas plus complexe que si une plate forme seule était utilisée.

1.2.6 Domaine d'application

L'objectif principal des drones est de remplir une mission qui pourrait être de nature militaire, scientifique, économique ou même commerciale. L'intérêt pour le contrôle et la navigation des drones est dû à leur utilisation dans des environnements dangereux [58].

Applications militaires (marine, armée et force aérienne)

- Intelligence électronique,
- Brouillage et destruction du système radar,
- Relais de signaux radio,
- Observation des flottes ennemies,
- Désignation et suivi des cibles,
- Élimination des bombes qui non pas explosées,

— Leurre de missiles par émission de signatures artificielles.

Applications civil

— Topographie aérienne pour les recherches géographiques,

— Pulvérisation et surveillance agricoles,

— Chercher et sauver,

— Lutte contre l'incendie et détection d'incendie forestier,

— Surveillance des importations illégales,

— Études de pollution et surveillance des terres,

— Inspection des pipelines et des lignes électriques,

— Recherche de pétrole et de gaz,

— Livraison de colis,

— Détection de véhicules mobiles au sol.

(A) Agriculture

(B) Protection civil

(C) Militaire

(D) Les désastres naturels

FIGURE 1.1 – L'utilisation des drones

1.3 Multi-drones coopérative et distribué pour la surveillance

Un troupeau est un mouvement collectif d'entités individuelles en interaction. Ce comportement est constamment observé dans la nature, des petites migrations d'insectes aux grands mammifères tels que les éléphants, et évidemment les oiseaux. Ce phénomène peut même être observé dans les microorganismes. La capacité de ces entités, normalement de faible intelligence, qui ne communiquent qu'avec leurs plus proches voisins, à se positionner correctement pour se déplacer dans un collectif, sans collision et assurant la

sécurité du groupe, a éveillé l'intérêt de divers chercheurs au cours des dernières décennies. Plusieurs modèles d'une grande pertinence ont émergé de ces enquêtes[14].

Les drones sont devenus un sujet brûlant au cours de la dernière demi-décennie pour plusieurs raisons :

— Il existe un fort intérêt commercial pour la production et la vente de petits drones qui donnent une satisfaction considérable aux clients.

— Les drones ont le potentiel d'être utilisés pour un certain nombre de fins pratiques, y compris l'agriculture, la surveillance, la réalisation de séquences aériennes et bien d'autres.

— Les drones sont utilisés pour combattre le terrorisme et ont d'autres applications militaires.

— Il existe une branche de l'industrie du divertissement en croissance rapide qui implique des dizaines de centaines de drones fournissant des effets visuels uniques en trois dimensions sur une échelle de centaines de mètres.

— Enfin, des volées de drones coopérants de manière autonome sont susceptibles d'ouvrir de nouvelles voies importantes à la fois concernant les points ci-dessus ainsi que leur utilisation pour résoudre des tâches complexes nécessitant de nombreuses unités «intelligentes».

1.3.1 Terminologies

Système multi-drones coopérative

Un système multi-UAV peut fonctionner de manière centralisée ou décentralisée. Dans un système centralisé, une entité sur le terrain recueille des

informations, prend des décisions pour les véhicules et met à jour la ou les missions. Dans un système décentralisé, les UAV doivent coopérer explicitement à différents niveaux pour atteindre les objectifs du système et échanger des informations pour partager les tâches et prendre des décisions collectives. Qu'il soit centralisé ou décentralisé, ce qui transforme un groupe d'UAV unique en un système multi-UAV est la coopération implicite ou explicite entre les véhicules [97]. Les drones doivent notamment :

- ✓ Observer leur environnement.
- ✓ Évaluer leurs propres observations ainsi que les informations reçues d'autres Les drones et leur raison.
- ✓ Agir de la manière la plus efficace.

Système multi-drones distribué

Les drones sont équipés d'appareils de radiocommunication et s'appuient sur des programmes de contrôle de vol autonomes sans pilote, qui ont été activement développés autour du monde. Compte tenu de leur faible coût, de leur capacité de manœuvre flexible et un fonctionnement sans pilote, les drones ont été largement utilisés à la fois dans les opérations civiles et les missions militaires,y compris la cartographie aérienne, le sauvetage en cas de catastrophe, l'agriculture, l'irrigation, etc.

Essaim de drones

Le vol d'essaimage est un nouveau domaine d'intelligence artificielle bio-inspirée basé sur les modèles comportementaux du vol d'essaim d'oiseaux, d'insectes et les autres animaux dans la nature. L'intelligence en essaim est définie comme un comportement collectif complexe, coordonné, flexible, auto-organisé et robuste d'un groupe d'individus qui suit une règle simple [8].

Le vol d'essaimage peut être utilisé par de nombreuses espèces d'oiseaux. Le vol en formation des oiseaux présente de nombreux avantages aérodynamiques, ce qui s'applique davantage aux grands oiseaux [72]. En outre, il existe d'autres théories qui considèrent le vol d'essaimage comme un mécanisme d'adoption anti-prédateur des oiseaux, ce qui est plus approprié pour les petits oiseaux, et aussi il peut arriver d'utiliser les comportements d'autres animaux comme les loups et les lucioles.

FIGURE 1.2 – Essaim de drones

Le tableau 1.2 montre une petite comparaison entre une architecture basée sur un seul drone et une autre basée sur la notion de multi-drones.

1.3.2 Architecture de communication

en trouve en générale trois différentes architectures pour les systèmes multi-drones :

Caractéristiques de comparaison	multi-drones	Un seul drone
Zone ciblée couverture	Plusieurs drones peuvent couvrir de plus grandes zones par rapport aux drones uniques	Les drones simples ne peuvent couvrir que des zones plus petites que les drones multi-drones
Coût	76/5000 Les coûts de maintenance d'un mini multi-UAV sont inférieurs au coût d'un gros drone	Les gros drones peuvent ne pas exiger des coûts de maintenance élevés.
Temps de traitement	La tâche requise peut être accomplie plus rapidement avec les multi-drones	La réalisation de la tâche requise est plus lente avec un seul drone.
Coupe transversale radar	Nécessite une section transversale radar plus petite	Nécessite une plus grande section transversale du radar
Puissance	Étant donné que le système multi-drones est construit à l'aide de petits UAV, les UAV de ces systèmes sont limités en puissance par rapport aux grands systèmes d'UAV uniques	La puissance des grands systèmes UAV uniques est efficace par rapport à celle des mini drones multi-UAV.
Topologie du réseau	Exigences de topologie de réseau complexes	Nécessite une connexion réseau directe et simple
Application	Large gamme d'applications	Gamme limitée d'applications
Sécurité	Plusieurs systèmes d'UAV ont de grandes surfaces d'attaque, en particulier lorsqu'ils font partie d'un système Internet des objets (IoT)	En général, les systèmes UAV uniques sont moins vulnérables que les systèmes multi-drones. Cependant, ils fournissent un point de défaillance unique en cas d'attaques réussies.

TABLE 1.2 – Multi-drones vs Mono-drone

Architecture centralisée : L'utilisation d'une architecture centralisée (figure 1.3) dont le donneur d'ordre a une connaissance globale de la flotte permet d'optimiser les actions de chaque élément de l'essaim. Néanmoins, ce type d'architecture nécessite le maintien d'un lien de communication entre l'entité de contrôle et au moins un des drones. La rupture de ce lien entraînerait un arrêt prématuré de la mission. La perte de communication entre le drone relais et les autres plate-formes conduiraient au même résultat. Plus le nombre de drones pour lesquels il faut calculer les actions est important, plus la quantité de calculs à réaliser est élevée, tout comme la quantité d'informations à transmettre à l'essaim. L'entité de contrôle doit donc disposer de ressources suffisantes [16] [70].

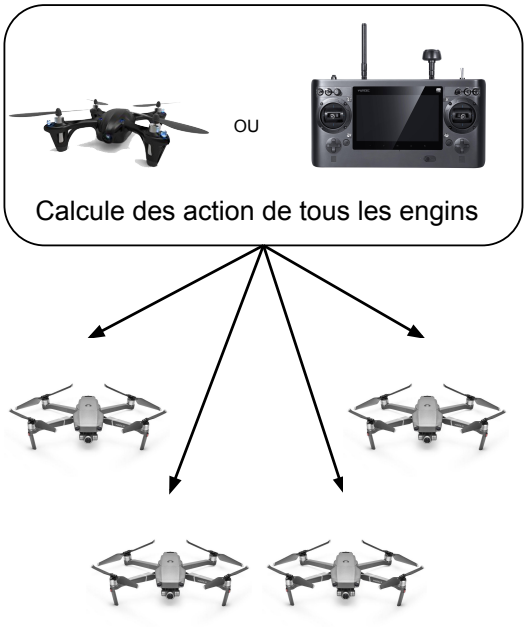


FIGURE 1.3 – Architecture centralisée

Architecture hiérarchique : L'architecture hiérarchique (figure 1.4)utilise plusieurs niveaux de contrôle de la flotte. On peut la considérer comme une

dérivée de l'architecture centralisée. Bien que l'architecture hiérarchique soit plus tolérante à l'extension de la flotte, elle peut entraîner des défaillances du système en cas de faille des entités de contrôle, notamment celles de haut niveau [16][70].

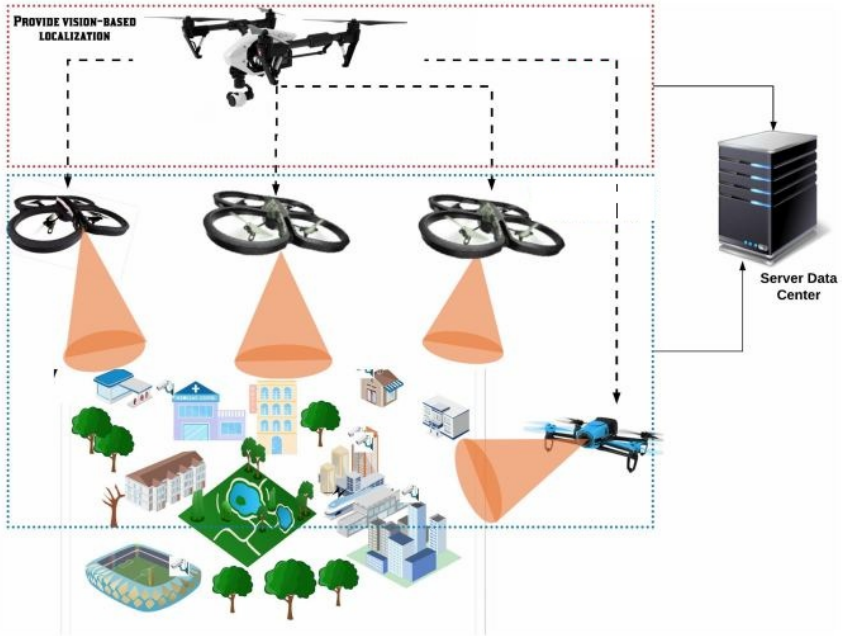


FIGURE 1.4 – Architecture hiérarchique

Architecture distribuée : Au sein d'une architecture distribuée (figure 1.5), chaque entité est autonome. Bien que les drones puissent échanger des informations utiles à leurs propres calculs, chacun dispose des capacités nécessaires à son auto-gestion. Ce type d'architecture permet l'extensibilité de la flotte, car elle ne nécessite ni le maintien de liens de communication ni l'utilisation de quelques entités pour des calculs intensifs. Enfin, l'architecture distribuée offre une forte tolérance aux pannes [16] [70].

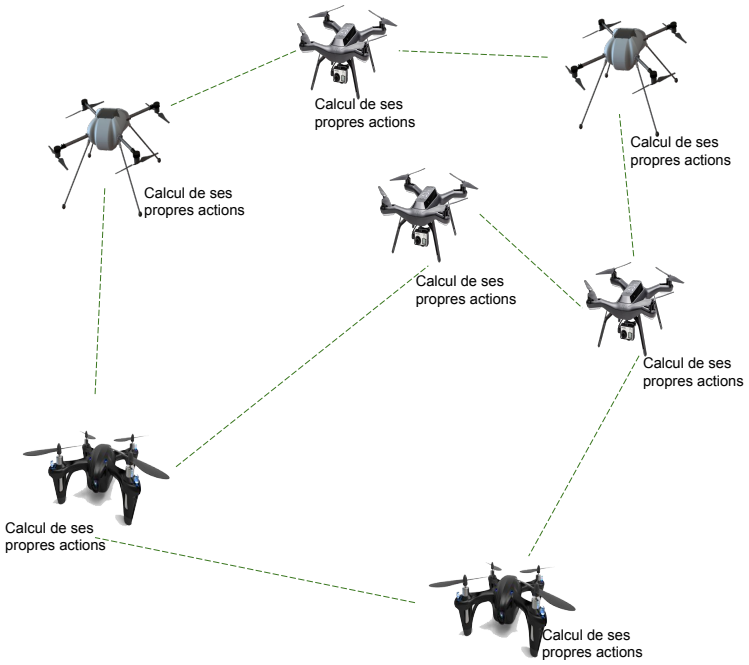


FIGURE 1.5 – Architecture distribuée

1.3.3 Les réseaux ad-hoc mobiles et véhiculaires

Les réseaux mobiles ad hoc classiques (MANET) et les réseaux ad-hoc véhiculaires (VANET) ont eu une grande influence sur les technologies de routage offrant ainsi la mobilité et l'agilité de vol aux systèmes UAV. Zhou et al.[99] ont proposé une architecture réseau coopérative aério-sol à deux couches où plusieurs drones formant le sous-réseau aérien assiste le sous-réseau véhiculaire terrestre grâce aux communications UAV-UAV et UAV-sol. Les drones agissent comme des relais intermédiaires en raison de leur mobilité flexible. Le système multi-drones a été proposé pour la première fois sur la base du concept FANET [6], où la méthodologie centrée sur le réseau a fourni aux drones la capacité de se positionner de manière autonome pour une connectivité idéale et une coopération avec d'autres drones afin d'obtenir

la meilleure couverture.

1.4 Quelques exemples de missions de surveillance basée sur un système multi-drones

— Désastre naturel : il y a (Salil Goel)[46] qui ont fait des études sur les technologies de communication et de réseau qui contribuent aux systèmes de gestion des drones dans les catastrophes , la recherche et le sauvetage, la communication d'urgence et la logistique. Pour démontrer les avantages de la communication d'urgence, M. Quaritsch et al. citequaritsch2010networked ont travaillé sur les capteurs des drones pour augmenté le rendement du drone pendant les désastres.

— L'agriculture : (Marek Kulbacki ; Jakub Segen ; Wojciech Knieć) ont développé des opérations agricoles sélectionnées telles que l'application à taux variable (VRA) basée sur des cartes ou des capteurs[42].

— Le suivi des objets : il y a les travaux Alexander C. Woods and Hung M. qui ont fait un travail sur le suivi des cibles dynamiques à l'aide d'un drone avec un coût réduit. Ce travail peut traquer des objets avec des positions et vitesses relatifs[87]. Idem pour Mouhyemen Khan, Karel Heurtefeuxet Amr Mohamed qui ont fait un système qui traque et suit les objets en mouvement avec un groupe de drones en utilisant le clustering [41].

1.5 Conclusion

Dans ce chapitre, nous avons présenté en détail les composants d'un drone, ses caractéristiques techniques et son domaine d'application. Nous avons

également abordé les systèmes multi drones en termes de coopération et de distribution des tâches. Nous avons présenté les différentes architectures de communication et nous avons terminé ce chapitre par un état de l'art sur quelques systèmes de surveillance basés sur une plate-forme multi drones.

CHAPITRE 2

LES ALGORITHMES MÉTAHEURISTIQUES BIO-INSPIRÉS CONCEPTS ET MÉTHODES

2.1 Introduction

101 machine (ordinateur) et tous ça par l'algorithmique, donc si en peut résoudre un problème simple il suffit d'utiliser un algorithme simple (des conditions, des boucles, des fonctions) mais si en veut résoudre un problème complexe qui contient beaucoup des paramètres et beaucoup des contraintes a respecter en trouve que les algorithmes simples ne fonctionne pas. donc la question est comment résoudre un problème complexe ? Un problème complexe peut être résolut par les algorithme bio-inspiré , un algorithme bio-inspiré est un algorithme qui contient une fonction s'appeler fonction objective et c'est lui qui le fait converger vers un optimum global avec le temps (plusieurs itérations) et trouve la solution optimale ou un peu optimale.

La résolution des problèmes d'optimisation complexes dans un espace de recherche qui augmente de manière exponentielle est devenue plus difficile

en utilisant des algorithmes d'optimisation classiques ou traditionnels en raison de son incapacité à obtenir des résultats satisfaisants. En cherchant des solutions de plus en plus performantes pour ces problèmes, les chercheurs ont opté pour une autre voie, c'est bien la nature. Cette dernière était depuis longtemps une source d'inspiratrice des chercheurs et des poètes dans leurs œuvres. Jusqu'à présent, la nature a résolu divers problèmes difficiles au cours de millions, voire de milliards d'années d'existences. Cependant, seules les meilleures solutions avec une certaine robustesse demeurent en suivant la loi de la survie au plus fort [90].

Ces dernières décennies, les phénomènes physiques ainsi que les comportements alimentaires et socioculturels des insectes et des animaux ont été largement observés et étudiés par les humains pour mieux comprendre le fonctionnement de la nature afin d'être capables de la gérer et s'adapter à ces conditions. Cette adaptation a permis de créer de nouveaux aliments en modifiant génétiquement des graines et des animaux et a contrôlé la propagation des maladies à l'aide de vaccins [26].

À la fin des années 80, de nouveaux algorithmes ont été développés sur la base du comportement évolutif de quelques systèmes naturels dans le but de résoudre des problèmes d'optimisation difficiles. Dans la littérature, ces derniers sont connus sous le nom d'algorithmes d'optimisation inspirés de la nature, métaheuristiques ou bio-inspirés. Ces algorithmes proviennent de la physique (recuit simulé, la recherche Tabou, etc.), de l'évolution biologique (algorithmes génétiques) ou de l'éthologie (les colonies de fourmis, l'essaim

particulaire, etc.) [50]. L'objectif des algorithmes méta-heuristique est de minimiser/maximiser une fonction objective qui décrit le degré d'approximation de la solution exacte en respectant une certaine vitesse relative au problème pour trouver un optimum et éviter les blocages dans un minimum/-maximum local [32].

Dans ce chapitre, nous présentons un état de l'art qui montre quelques détails techniques sur les métaheuristiques bio-inspirés. Nous présentons d'abord en détail les concepts et définitions liés à la métaheuristique bio-inspirée et à l'intelligence des essaims. Ensuite, nous allons faire un tour d'horizon sur les algorithmes métaheuristiques bio-inspirés proposés dans la littérature. Nous consacrons le reste du chapitre à une description détaillée des métaheuristiques bio-inspirés utilisées dans ce mémoire.

2.2 Définition et Caractéristiques

"Heuristique" est un mot grec qui signifie "savoir", "trouver, "découvrir" ou "guider une enquête" [45]. Selon Russell Norvig, les heuristiques sont des techniques qui recherchent de bonnes solutions (quasi-optimales) à un coût de calcul raisonnable sans pouvoir garantir la faisabilité, ou même dans de nombreux cas, indiquer à quel point une solution réalisable particulière est proche de l'optimale [68].
Pour mieux comprendre la méta-heuristique, il est important de définir ses termes fondamentaux. Les définitions suivantes sont utiles à cet effet :

Définition 5 *Une métaheuristique est formellement définie comme un processus de génération itératif qui guide une heuristique subordonnée en combinant intelligemment différents concepts pour explorer et exploiter l'espace de recherche. Plusieurs stratégies d'apprentissage sont utilisées pour structurer les informations afin*

de trouver efficacement des solutions quasi optimales (Osman, I.H. and Laporte, G. [59]).

Définition 6 *Une méta-heuristique est un processus itératif qui guide et modifie les opérations des heuristiques subordonnées pour produire efficacement des solutions de haute qualité. Il peut manipuler une solution unique complète (ou incomplète) ou un ensemble de solutions à chaque itération. Les heuristiques subordonnées peuvent être des procédures de haut (ou bas) niveau, ou une simple recherche locale, ou simplement une méthode constructive. (Voß, Stefan, et al. [82]).*

Définition 7 *Une métaheuristique est un ensemble de concepts qui peuvent être utilisés pour définir des méthodes heuristiques pouvant être appliquées à un large ensemble de problèmes différents. En d'autres termes, une métaheuristique peut être considérée comme un cadre algorithmique général qui peut être appliqué à différents problèmes d'optimisation avec relativement peu de modifications pour les adapter à un problème spécifique (Metaheuristics Network Website [51]).*

Définition 8 *Les métaheuristiques sont généralement des stratégies de haut niveau qui guident une heuristique sous-jacente plus spécifique à un problème afin d'augmenter leurs performances. L'objectif principal est d'éviter les inconvénients de l'amélioration itérative et, en particulier, de la descente multiple en permettant à la recherche locale d'échapper à l'optima local. Pour ce faire, il faut soit permettre des mouvements de plus en plus graves, soit générer de nouvelles solutions de départ pour la recherche locale d'une manière plus "intelligente" que la simple fourniture de solutions initiales aléatoires. De nombreuses méthodes peuvent être interprétées en introduisant un biais pour garantir une production rapide de solutions de haute qualité. Ce dernier peut prendre diverses formes et peut être exprimé comme un biais de descente (basé sur la fonction objective), un biais de mémoire (basé sur des décisions prises antérieurement) ou un biais d'expérience (basé sur des performances*

antérieures). De nombreuses approches métaheuristiques reposent sur des décisions probabilistes prises pendant la recherche. Mais la principale différence avec la recherche aléatoire pure est que dans les algorithmes métaheuristiques, le caractère aléatoire n'est pas utilisé à l'aveuglette, mais sous une forme intelligente et biaisée (Stützle, T. [75]).

Définition 9 *Une métaheuristique est un algorithme d'optimisation visant à résoudre des problèmes d'optimisation difficile (souvent issus des domaines de la recherche opérationnelle, de l'ingénierie ou de l'intelligence artificielle) pour lesquels on ne connaît pas de méthode classique plus efficace. Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est-à-dire l'extremum global d'une fonction, par échantillonnage d'une fonction objectif. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution (d'une manière proche des algorithmes d'approximation)(Wikipédia [56]).*

En terme général, une métaheuristique est une méthode algorithmique capable de guider et d'orienter le processus de recherche dans un espace de solution, souvent très grand à des régions riches en solutions optimales. Le fait de rendre cette méthode abstraite et plus générique conduit à une vaste utilisation pour différents champs d'applications . Avec ces applications, les métaheuristiques permettent de trouver des solutions pas toujours optimales, mais proches de l'optimum et déraisonnable [4].

2.3 Motivation de l'utilisation

Les algorithmes métaheuristiques sont utilisé dans divers domaine grâce a :

- ✓ **La flexibilité** : par leur capacité à traiter de nombreuses applications avec des exigences diverses. Elles permettent de gérer le compromis entre performance et qualité des solutions. Cela implique un lien fort entre la métaheuristique et les applications de la vie réelle [79].

- ✓ **La simplicité** : la majorité des métaheuristiques sont faciles à comprendre (en possédant un certain niveau) à cause de leurs origines inspirées du comportement des êtres vivants ; donc il suffit de connaître comment ils interagissent [35].

- ✓ **Large utilisation** : les métaheuristiques ont plusieurs méthodes (bio-inspirée, algorithmes évolutionnaires, etc.), chacune d'elle est efficace pour un certain type de problème ; donc, il suffit de comprendre le problème et d'utiliser la méthode adéquate [85].

- ✓ **La vitesse** : les algorithmes métaheuristiques garantissent généralement la vitesse d'exécution si la fonction objectif est bien définie, mais ils ne garantissent pas la solution exacte [57].

- ✓ **La puissance** : les métaheuristiques se montrent plus puissantes que les méthodes de parcours exhaustif ou de recherche purement aléatoire.[57]

- ✓ **La décentralisation** : cette propriété garantit un système robuste, capable de continuer à fonctionner même en cas de défaillance de l'un de ses composants.

- ✓ **La réactivité** : les éléments d'un système qui se base sur la métaheuristique coopèrent et communiquent entre eux par le biais d'interactions

locales. Ils sont capables de réagir instantanément aux changements de l'environnement.

2.4 Intelligence en essaim

Le terme « intelligence en essaim » ou en anglais « Swarm Intelligence », a été créé par Gerardo Beni en 1989 : « *L'intelligence en essaim est une propriété des systèmes de robots non intelligents présentant un comportement intelligent collectif* »[7].

L'intelligence en essaim fait référence à une sorte de capacité émergente de la résolution de problèmes avec des interactions de simples unités de traitement de l'information. Le concept d'un essaim suggère la multiplicité, la stochastique, le caractère aléatoire et le désordre. Le concept de l'intelligence suppose en amont que la méthode de résolution de problèmes est en quelque sorte réussie. Les unités de traitement de l'information qui composent un essaim peuvent être animées, mécaniques, informatiques ou mathématiques ; ils peuvent être des insectes, des oiseaux ou des humains ; il peut s'agir d'éléments de réseau, de robots ou de postes de travail autonomes ; ils peuvent être réels ou imaginaires. Leur couplage peut avoir un large éventail de caractéristiques, mais il doit être une interaction entre les unités. Étant donné la diversité des paradigmes, ce chapitre se concentrera sur l'approche particulière connue sous le nom d'optimisation des essaims de particules [24].

2.4.1 Avantages

L'intelligence en essaim est un domaine émergent de l'intelligence artificielle d'inspiration biologique basée sur les modèles comportementaux d'insectes sociaux tels que les fourmis, les abeilles, les guêpes, les termites, etc. [76]. il devient un domaine très intéressant grâce à ses avantages qu'on les détermine comme suit :

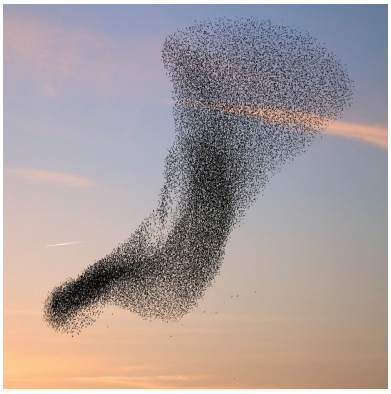
- ✓ **Flexible :** La colonie réagit aux perturbations internes et aux défis externes.
- ✓ **Robuste :** Les tâches sont terminées même si certains agents échouent.
- ✓ **Évolutif :** De quelques agents à plusieurs millions.
- ✓ **Décentralisé :** Il n'y a pas de contrôle central dans la colonie.
- ✓ **Auto-organisé :** Les solutions sont émergentes plutôt que prédéfinies.
- ✓ **Adaptation :** Le système d'essaim peut non seulement s'adapter à des stimuli prédéterminés, mais également à de nouveaux stimuli.
- ✓ **Vitesse :** Les changements dans le réseau peuvent se propager très rapidement.
- ✓ **Modularité :** Les agents agissent indépendamment des autres couches du réseau.
- ✓ **Parallélisme :** Les opérations des agents sont intrinsèquement parallèles.

La figure 2.1 montre quelques exemples naturels des essaimes utilisés dans les méthaheuristiques.

(A) Essaim de poissons

(B) Essaim d'abeilles

(C) Essaim de fourmis



(D) Essaim des oiseaux

FIGURE 2.1 – Essaims intelligents

2.5 Une brève revue

Au cours des dernières décennies, il y a eu un intérêt croissant pour les algorithmes inspirés par les comportements des phénomènes naturels [93, 67, 60, 65, 52, 69, 9]. De nombreux chercheurs ont montré que ces algorithmes sont bien adaptés pour résoudre des problèmes complexes [44]. Dans l'enquête [21], Dokeroglu et al. ont étudié les nouveaux algorithmes métaheuristiques qui ont été introduits au cours des vingt dernières années (entre 2000 et 2020). Dans cette étude, les algorithmes ont été divisés chronologiquement

en deux grandes générations.

La première génération d'algorithmes métaheuristiques nommée *la génération classique* comprend presque tous les algorithmes qui ont été développés avant l'an 2000. Cette génération est caractérisée par un grand nombre de nouvelles métaheuristiques inspirées par des processus évolutifs ou comportementaux. Cette vague d'approches métaheuristiques apporte les meilleures solutions pour certains nombres de problèmes de référence non résolus [21]. Les algorithmes classiques selon Dokeroglu et al. [26] sont mentionnés ci-dessus :

Algorithmes génétiques (GA) [30], Optimisation des essaims de particules (PSO) [83], Optimisation des colonies de fourmis (ACO) [22], Programmation génétique (GP) [5], Evolution différentielle (DE)[63], Recuit simulé (SA) [81], Recherche tabou (TS)[29], Procédure de recherche adaptative aléatoire gloutonne (GRASP) [49], Algorithme immunitaire artificiel (AIA) [18], Recherche locale itérée (ILS) [74], La méthode d'optimisation du chaos (COM) [10], La recherche par dispersion (SS)[48], L'algorithme du saut de grenouille mélangé (SFLA) [34], et La recherche de voisinage variable (VNS) [55].

Tandis que la deuxième génération, connue sous le nom de *la nouvelle génération* d'algorithmes métaheuristiques comprend les algorithmes qui ont été développés après l'an 2000 jusqu'à aujourd'hui. Les algorithmes métaheuristiques de nouvelle génération qui ont été détaillés et discutés dans [21] sont mentionnés ci-dessus :

Colonie d'abeilles artificielles (ABC) [37], Recherche de nourriture bactérienne (BFO) [17], Algorithme des chauves-souris (BA) [89], Optimisation

basée sur la biogéographie (BFO) [73], Recherche de coucou (CS) [94], Algorithme de la firefly (FA) [91], Algorithme de recherche gravitationnelle (GSA) [67], Algorithme du loup gris (GWO)[54], Recherche d'harmonie (HS) [28], Troupeau de krill (KH) [27],Optimisation de Social Spider (SSO)[15], Recherche d'organismes symbiotiques (SOS) [13], L'enseignement de l'optimisation basée sur l'apprentissage (TLBO) [66], et Algorithme d'optimisation des baleines (WOA) [53].

La plupart des algorithmes métaheuristiques sont inspirés de la nature, le comportement animal ou des phénomènes physiques [39, 40]. Mirjalili et Lewis [53] classent les algorithmes métaheuristiques en trois classes : Les méthodes basées sur l'évolution, sur la physique et sur les essaims. Les méthodes basées sur l'évolution imitent le processus d'évolution dans la nature pour effectuer l'optimisation. Alors que les algorithmes basés sur la physique effectuent l'optimisation en utilisant les règles de la physique dans l'univers. La troisième classe d'algorithmes métaheuristiques est constituée par les techniques basées sur les essaims, qui imitent le comportement des animaux dans un groupe. Le tableau 2.1 présente une vue schématique de la classification des algorithmes métaheuristiques.

2.6 Recherche de l'Aigle

2.6.1 Strategie

La stratégie de recherche de l'aigle a été développée par Xin-She Yang et Suash Deb [95]en 2010. Cette stratégie a été inspirée par le comportement de

TABLE 2.1 – Quelques algorithmes métaheuristiques

Algorithmes métaheuristiques		
Évolutifs	Basés sur la physique	Basés sur les essaims
Algorithmes génétiques (GA) [19]	Recuit simulé (SA)	Optimisation de l'essaim de particules (PSO)
Stratégie d'évolution (ES)[61]	Algorithme de recherche gravitationnelle (GSA)[67]	Optimiseur de loup gris (GWO) [54]
Optimiseur basé sur la biogéographie (BBO)[98]	Optimisation des forces centrales (CFO)	Optimisation de la flamme des papillons de nuit (MFO)
Programmation évolutive (EP)[31]	Algorithme des trous noirs (BH)	Algorithme d'optimisation des baleines (WOA)
	Algorithme de recherche basé sur les galaxies (GBSA)	Colonie d'abeilles artificielles (ABC) [38]
	Optimisation de l'espace courbe (CSO)	Recherche de coucou (CS) [65]
	Algorithme d'optimisation magnétique (MOA)	Algorithme de firefly (FA)[91]
	Algorithme d'optimisation en spirale (SOA)	Algorithme de vol de lèvy (LF)
	Algorithme du cycle de l'eau (WCA)	Algorithme de pingouin (EPO)[33]

recherche de nourriture d'aigles tels que les aigles royaux ou l'Aquila Chrysaetos [95]. Selon les auteurs, la stratégie de la recherche d'Aigle est une méthode d'optimisation en deux étapes comme le montre l'algorithme 1. La première s'appelle une recherche globale dans laquelle l'aigle utilise une vision globale pour balayer la surface à la recherche d'une proie attendue. Lorsque ce dernier trouve une proie prometteuse, il change sa stratégie en une stratégie de chasse. Dans la deuxième stratégie, appelée recherche locale, l'aigle utilise une stratégie de recherche focalisée autour de la zone dans laquelle il a observé la proie (recherche globale).

Algorithme 1 : Stratégie de l'Aigle [96]

Fonctions objectives $f_1(x), ..., f_N(x)$

Première estimation $x^{t=0}$

while *(Critres d'arrt non satisfaits)* **do**

  L'exploration globale par la randomisation

  Évaluer les objectifs et trouver une solution prometteuse

  Recherche locale intensive autour d'une solution prometteuse via un optimiseur local efficace

  if *(une meilleure solution est trouve)* **then**

    Mettre à jour la meilleure actuelle

  end

  Mettre à jour $t = t + 1$

end

Résultats du post-traitement et visualisation

L'avantage d'une telle combinaison est de créer un certain compromis entre la recherche globale, généralement lente, et une recherche locale rapide. Un autre avantage de cette méthode est que on pouvons utiliser tous les algorithmes que nous voulons à différents stades de la recherche ou même à différents stades des itérations. Il est donc facile de combiner les avantages

de divers algorithmes afin de produire de meilleurs résultats [96].

Basé sur l'hypothèse que l'aigle fouille son propre territoire en volant librement et de manière aléatoire, tout comme les vols de Lévy. Xin-She Yang et Suash Deb [95]supposent que la marche aléatoire du vol de Lévy est la plus appropriée pour la première stratégie de l'Aigle. Pour la seconde stratégie, les auteurs recommandent l'utilisation n'importe quel algorithme métaheuristique rapide et efficace tel que l'optimisation des essaims de particules (PSO) ou l'algorithme Firefly (FA).

2.6.2 Recherche globale (Lévy Flight)

L'algorithme de Lévy Flight a été introduit par un mathématicien français en 1937 nommé Paul Lévy (1886-1971) [36]. Cet algorithme a été utilisé presque par tout :le marketing, la cryptographie, l'astronomie et même dans les réseaux sociaux [47]. Lévy Flight est un type de marche aléatoire dans lequel les pas sont tirés d'une distribution de Lévy. En général, Lévy Flight est un ensemble de mouvements courts aléatoires liés par de rares mouvements plus longs (voir la Figure 2.2).Voici quelques exemples de Lévy Flight :

- ✓ Un requin qui se nourrit restera dans une petite zone, à la recherche de poisson. Ensuite, il se rendra compte qu'il a utilisé toutes les sources à cet endroit. Il se dirigera ensuite dans une direction aléatoire, parcourra une certaine distance et recommencera à chercher de la nourriture [47].

- ✓ Quelqu'un découvre un site Web. Il le visite une fois, puis revient et revient encore. Ensuite, il sent qu'il n'y a plus d'avantages et il continue sa navigation jusqu'à ce qu'il trouve un autre site Web à visiter [47].

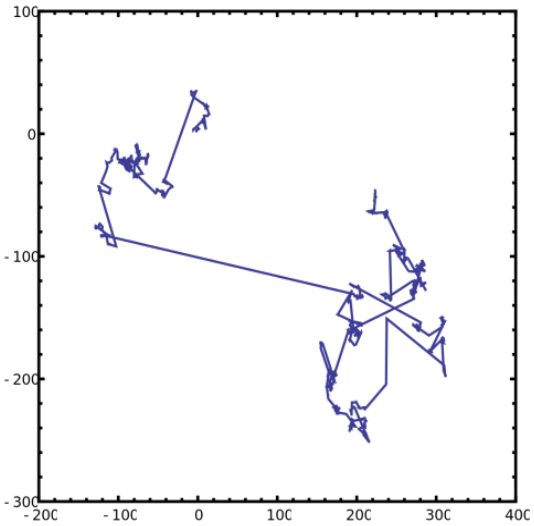


FIGURE 2.2 – Simulations de traces de Lévy Flight.

L'algorithme Lévy Flight est vraiment utile pour la recherche aléatoire, la longueur des pas est définie par une distribution de probabilité basée sur l'équation suivante :

$$P_{\alpha\gamma}(x) = \frac{1}{\pi}\int_a^b e^{-y}cos(qx)dq \tag{2.1}$$

$$y = \alpha q^{\alpha} \tag{2.2}$$

Cette distribution est symétrique pour x=0, avec α est le facteur de contrôle de la distribution compris entre 0 et 2, et γ est le facteur d'échelle. On fixe γ=1, la distribution de probabilité peut être approximé par :

$$P_{\alpha}(x) \approx x^{-\alpha-1} \tag{2.3}$$

La longueur des pas l est générer par l'équation suivante :

$$l = l_0(\frac{1}{\frac{1}{\beta\alpha}}) - 1 \tag{2.4}$$

Où, l_0 est la longueur initiale et $\beta \in$[0,1].

L'algorithme 2 présente le pseudo code de l'algorithme de Lévy Flight.[36]

Algorithme 2 : L'algorithme de Lévy Flight [36]

```
int_position()
curr_val ← fonction_objective()
best_val ← curr_val
best_position ← curr_position()
while Critres d'arrt non satisfaits do
    while saut non pas atteindre le maximum do
        vol_len ← Lvy_Flight(base_len, beta)
        sautent aléatoirement à distance (vol_len)
        curr_val ← objective function()
        if curr_val<best_val then
            best_val ←curr_val
            best_position ← curr_position()
        end
    end
    return to best known position()
end
```

2.6.3 Recherche locale (Firefly)

L'algorithme de Fireflies a été développé par Xin-She Yang en 2008 [92]et il est basé sur les modèles de clignotement et le comportement des fireflys tropicales. Les fireflys (Fireflies) ont un système lumineux qui sert a réaliser plusieurs tâches comme la communication entre les fireflys, l'attraction des proies, et enfin repoussent les ennemis [88]. La **figure 2.3** détermine le comportement des lucioles.

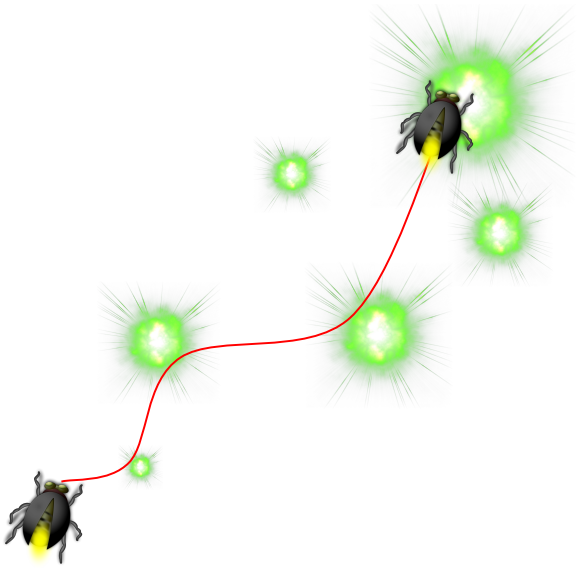


FIGURE 2.3 – Le comportement du Luciole

Les étapes de base de l'algorithme Firefly sont résumées sous la forme du pseudo code illustré dans l'algorithme 3, qui comprend les trois règles

suivantes [91] :

Algorithme 3 : Pseudo code de l'algorithme Firefly (FA) [91]

Fonction objective f(x), x = (x_1...x_d)
Initialiser une population de fireflies $x_i (i = 1, 2, ..., n)$
Définir le coefficient d'absorption de la lumière γ
while *(t < Génération_Max)* **do**
  for $i \leftarrow 1$ **to** *n fireflies* **do**
    for $j \leftarrow 1$ **to** *i fireflies* **do**
      L'intensité lumineuse I_i à x_i est déterminée par $f(x_i)$
      if $(I_j > I_i)$ **then**
        Déplacez Firefly i vers j dans toutes les dimensions d
      end
      L'attractivité varie avec la distance r par exp[-γr^2]
      Évaluer de nouvelles solutions et mettre à jour l'intensité lumineuse
    end
  end
  Classez les fireflies et trouvez le meilleur actuel
end
Résultat et visualisation du post-traitement

✓ Toutes les fireflies sont unisexes de sorte qu'une firefly est attirée par d'autres fireflies, quel que soit leur sexe.

✓ L'attractivité est proportionnelle à leur luminosité, donc pour deux fireflies qui clignotantent, la moins brillante se déplacera vers la plus brillante. L'attractivité est proportionnelle à la luminosité et ils diminuent tous les deux au fur et à mesure que leur distance augmente. Sinon l'un est plus brillant que la firefly particulière, il se déplace de façon aléatoire.

✓ La luminosité ou l'intensité lumineuse d'une firefly est affectée ou déterminée par le paysage de la fonction objectif à optimiser.

2.7 Algorithme des Loups Gris (GWO)

L'algorithme des loups gris est introduit par Mirjalili et Andrew Lewis en 2014 [54]. Il suit la hiérarchie des dirigeants et la politique de poursuite des loups gris. Le loup gris est un animal social qui vit dans un pack de 5 à 12 membres. L'ensemble du groupe peut être classé en quatre types de loups gris, qui sont alpha (α), bêta (β), delta (δ) et oméga (ω) [43].Alpha détient le plus haut niveau dans la hiérarchie sociale du groupe, qui diminue ensuite jusqu'à l'oméga. Alpha est le chef de meute. Il dicte ses décisions concernant la chasse, le lieu de sommeil, etc. Les bêtas, qui peuvent être des mâles ou des femelles, sont des loups subordonnés à l'alpha, l'un d'eux est probablement le meilleur candidat, qui remplace un alpha en cas de décès ou de vieillissement. Les bêtas sont responsables du maintien à la discipline dans la proie et le renforcement des commandes de l'alpha. En même temps, ils donnent un retour sur l'alpha. Omega a le rang le plus bas dans le classement du loup gris. Ce sont les derniers loups autorisés à manger [54].

Les loups gris ont un mécanisme bien défini pour la chasse. Au tout début, ils s'approchent de la proie par pistage et poursuite. Après cela, la meute de loups gris encercle la proie et la harcèle jusqu'à l'arrêt du mouvement. Après cela, ils attaquent vers la proie [54].

Le comportement d'encerclement de la meute de loups gris peut être défini à l'aide des équations mathématiques suivantes :

$$\overrightarrow{D} = \overrightarrow{C}.\overrightarrow{X_p}(t) - \overrightarrow{X}(t) \tag{2.5}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X_p}(t) - \overrightarrow{A}.\overrightarrow{D} \tag{2.6}$$

Où $\rightarrow X_p$ est le vecteur de position de la proie, tandis que $\overrightarrow{X}$ est le vecteur de position d'un loup gris. (t) indique l'itération actuelle, et $\overrightarrow{A}$ et $\overrightarrow{C}$ sont les vecteurs de coefficient, qui peuvent être calculés comme suit :

$$\overrightarrow{A} = 2\overrightarrow{a}.\overrightarrow{r_1} - \overrightarrow{a} \tag{2.7}$$

$$\overrightarrow{C} = 2.\overrightarrow{r_2} \tag{2.8}$$

Où $\overrightarrow{r_1}$ et $\overrightarrow{r_2}$ sont des vecteurs aléatoires dans [0, 1] et des composants de $\overrightarrow{a}$ diminuent linéairement de 2 à 0 au cours des itérations.
Dans la modélisation mathématique du comportement de chasse des loups gris, on suppose que l'alpha (meilleure solution candidate), la bêta et le delta ont une meilleure connaissance de l'emplacement des proies. C'est pourquoi nous avons enregistré les trois premières meilleures solutions obtenues jusqu'à présent et obligé les autres agents de recherche, y compris les omégas, à mettre à jour leurs positions en fonction de la position des meilleurs agents de recherche selon les équations suivantes[54] :

$$\overrightarrow{D_\alpha} = |\overrightarrow{C_1}.\overrightarrow{X_\alpha} - \overrightarrow{X}|,\ \overrightarrow{D_\beta} = |\overrightarrow{C_2}.\overrightarrow{X_\beta} - \overrightarrow{X}|,\ \overrightarrow{D_\delta} = |\overrightarrow{C_3}.\overrightarrow{X_\delta} - \overrightarrow{X}| \tag{2.9}$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}.(\overrightarrow{D_\alpha}),\ \overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2}.(\overrightarrow{D_\beta}),\ \overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3}.(\overrightarrow{D_\delta}) \tag{2.10}$$

FIGURE 2.4 – les différentes étapes de chasse

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \tag{2.11}$$

Afin de propager l'exploration et l'exploitation, le paramètre a passe de 2 à 0. Mathématiquement, $|A| > 1$ force les loups gris à s'écarter de la proie à la recherche d'une proie plus en forme (exploration), tandis que $|A| < 1$ force les loups à attaquer vers la proie.

Algorithme 4 : Pseudo-code de l'algorithme GWO [54]

Initialiser la population de loups gris $X_i (i = 1, 2, ..., n)$

Initialiser a, A, et C

Calculer l'aptitude est (fitness) de chaque agent de recherche

$X_\alpha \leftarrow$le meilleur agent de recherche

$X_\beta \leftarrow$le deuxième meilleur agent de recherche

$X_\delta \leftarrow$le troisième meilleur agent de recherche

while *(t < Nombre maximal d'itérations)* **do**

  for *chaque agent de recherche* **do**

    Mettre à jour la position de l'agent de recherche actuel par équation (2.11)

  end

  Mettre à jour a, A, et C

  Calculer l'aptitude est (fitness) de chaque agent de recherche

  Mettre à jour $X_\alpha, X_\beta, et X_\delta$

  t = t + 1

end

retourner X_α

2.8 Conclusion :

Nous avons présenté dans ce chapitre l'utilité des algorithmes inspirés de la nature pour résoudre les problèmes les plus complexes. En effet, au lieu de penser à nouveau, il suffit d'étudier les solutions proposées par la nature qui a des millions d'années d'existence avant nous les humains et qui a réussi dans cette longue période à booster au maximum les performances en matière de survie. Nous avons également détaillé quelques algorithmes

bio-inspirés qui seront utilisés dans l'élaboration de notre système dans le chapitre suivant.

CHAPITRE 3

SYSTÈME DE SURVEILLANCE BASÉ SUR UNE APPROCHE COOPÉRATIVE ET DISTRIBUÉ DES DRONES

3.1 Introduction

Dans les dernières années,il y a eu intérêt considérable pour le développement et l'amélioration des outils de surveillance soit dans les endroits public ou privé. Les véhicules aériens sans pilote (UAV) ou drone sont parmi les nouvelles technologies qui sont utilisées dans le contexte de la vidéo surveillance. Les drones sont des objets volants sans pilote à bord, ils sont équipés par divers capteurs (tels que des caméras) et ils possèdent une capacité de traitement qui permet un déploiement facile, une grande mobilité et un large champ de vision.

3.2 Problématique

Dans le contexte du suivi des objets mobile, un seul drone n'est pas suffisant pour fournir des taches opérationnelles complètes, en raison de leur limitation dans la capacité de traitement, l'efficacité et la qualité de la mission, la flexibilité dans l'exécution des tâches et le temps de vol (énergie).

Pour surmonter les limitations des drones dans les systèmes de surveillance, les récents travaux scientifiques ont mis l'accent sur la modélisation des essaims pour mieux comprendre comment des animaux sociaux interagissent et atteignent leurs objectifs (Swarm UAV). Ces derniers sont utilisés dans de nombreuses applications telles que l'exploration, la surveillance , les attaques et la cartographie d'environnements inconnus. Techniquement, un «UAV essaim » est un groupe d'UAV communiquant les uns avec les autres de manière autonome à l'aide d'un algorithme de contrôle intelligent embarqué. Dans le but de réduire le coût du système de surveillance en termes d'énergie, bande passante et capacité stockage et d'améliorer la qualité du suivi des objets mobiles, nous allons proposer un nouvel algorithme distribué basé sur les méta heuristiques et qui permet de gérer et coordonner des drones pour le suivi des objets en mouvement. Cependant, l'utilisation d'un groupe d'UAV va engendrer plusieurs problèmes dont il faut absolument trouver une solution. Parmi ces problèmes nous citons :

- ✓ Comment savoir le nombre de drones qu'on a besoin ?
- ✓ Quelle architecture de communication va-t-on utiliser ?
- ✓ Comment gérer les mouvements des drones ?
- ✓ Quel type d'algorithme faut-il choisir ?

✓ Comment remplacer un drone lorsque sa batterie est expirée ?

Donc, notre objectif est de concevoir un système de surveillance basé sur la coopération de plusieurs UAV en maximisant leurs performances et en diminuant au maximum leurs consommations d'énergie.

3.3 Conception du Système

3.3.1 Stratégie

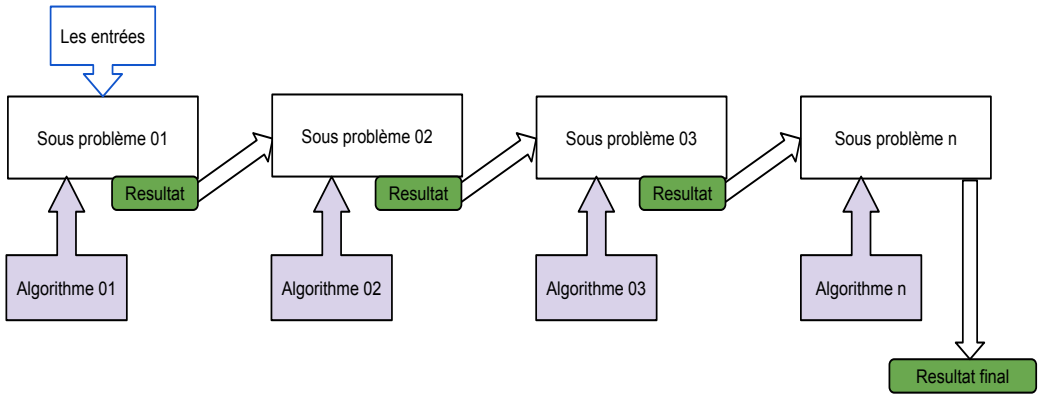


FIGURE 3.1 – Diviser pour résoudre

Avant de choisir un algorithme, il faut bien comprendre le problème (la gestion des drones dans notre cas),ensuite trouver une méthode à suivre pour résoudre ce problème. Parfois il suffit de choisir un seul algorithme pour résoudre un problème, mais dans notre cas, on a trouvé qu'un seul algorithme ne peut satisfaire nos besoins. Pour cela, on a décomposé le problème en sous-problèmes et on a choisi un algorithme pour traiter chaque sous-problème. Le résultat obtenu de chaque sous-problème va jouer le rôle de donnée en entée dans le sous-problème suivant et ainsi de suite. Cette

méthode s'appelle "diviser pour résoudre", c'est une méthode algorithmique basée sur le principe suivant :
On prend un problème (généralement complexe à résoudre), on divise ce problème en une multitude de petits problèmes, l'idée étant que les "petits problèmes" seront plus simples à résoudre que le problème original. Une fois les petits problèmes résolus, on recombine les "petits problèmes résolus" afin d'obtenir la solution du problème de départ[20]

3.3.2 Modélisation

3.3.3 Choix des algorithmes

Afin de mettre en oeuvre notre système, nous avons choisi un algorithme métaheuristique pour chaque module. En effet, chaque sou problème a besoin d'un algorithme bien spécifique pour le résoudre. Nous avons opté pour l'algorithme de l'aigle pour la recherche, l'algorithme des loups gris pour le suivi et l'encerclement du problème et enfin nous avons utilisé un algorithme simple pour le remplacement des drones qui ont consommé leur batterie.

L'algorithme de l'aigle : Tel qu'il a été présenté dans le chapitre 02, l'algorithme de l'aigle est divisé en deux étapes, chacune d'elle est gérée par un algorithme. Nous avons choisi d'utiliser "lèvy flight" pour la vision globale à cause de son pouvoir à bien traiter des problèmes aléatoires qui apparaissent de façon aléatoire dans des places aléatoires. L'algorithme "lèvy flight" n'a pas une stratégie de mémorisation des places visitées, pour cela nous avons augmenté la possibilité de visiter le même endroit plusieurs fois dans des périodes différentes. Pour la vision locale, nous avons choisi d'utiliser "Fireflies" qui concorde exactement avec nos besoins de traiter les problèmes dans

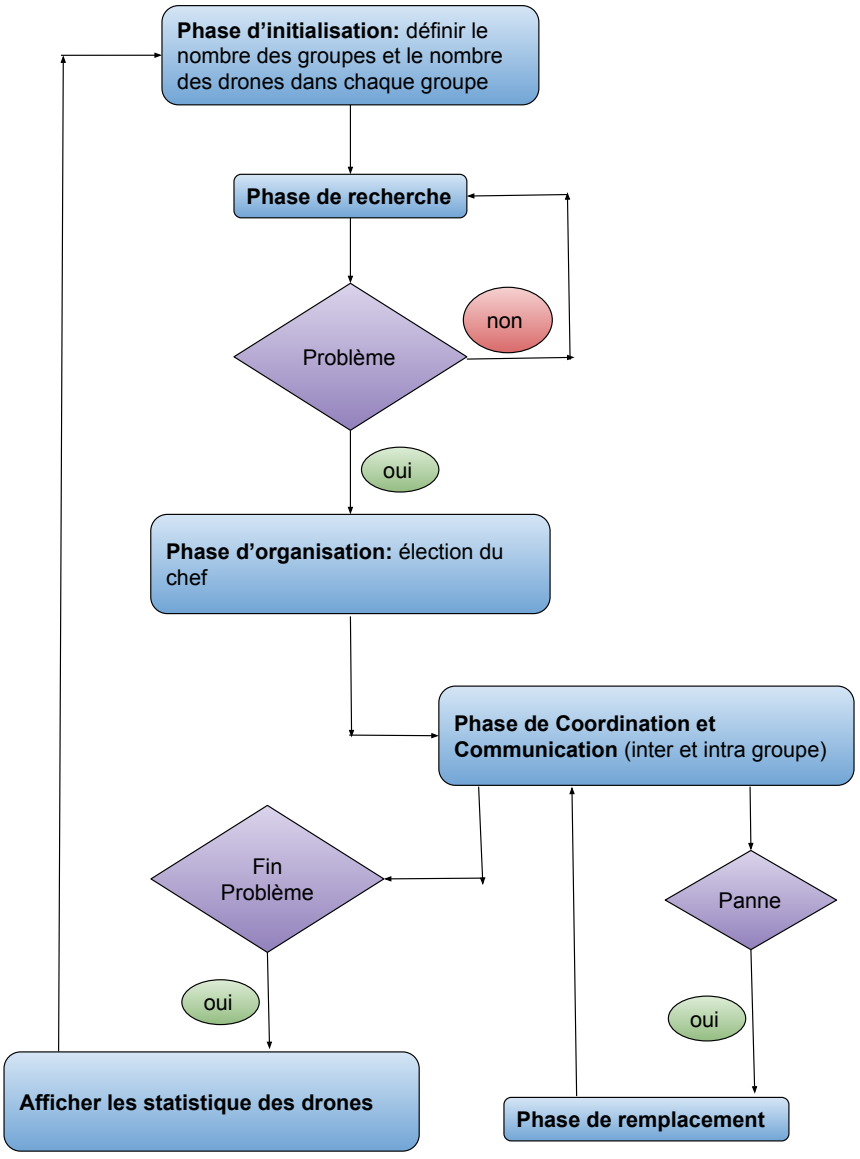


FIGURE 3.2 – Organigramme du système proposé

un ordre décroissant. En effet, on commence d'abord par traiter le grand problème et ainsi de suite si son emplacement lui permet bien sûr.

Algorithme des loups gris : Dans la tache de surveillance, nous avons besoin à un certain moment d'un drone chef qui s'occupe de la gestion des communications entre les drones sélectionnés pour la surveillance, de la stratégie de remplacement utilisé en cas de baisse de batterie de l'un des drones ; ou pour organiser les drones afin de couvrir plus de 80% du problème détecté. Cet algorithme est basé sur la sélection du chef et la modification dynamique des positions des loups par rapport à la position de ce dernier. Il est clair que cet algorithme fournit une modélisation qui peut être qualifiée de très compatible avec le problème de la gestion des drones dans le cas d'une détection de quelconque anomalie dans la surface de surveillance.

3.3.4 Phase d'initialisation

C'est la première phase, ici on va définir le nombre de drones et aussi le nombre de problèmes, chaque drone va commencer par une batterie pleine (100%)

3.3.5 Phase de recherche (Eagle Search)

La phase de recherche est assurée par l'algorithme "Eagle Search". ES est un algorithme qui s'exécute à deux phases (chapitre 02) : une pour la recherche globale et l'autre pour la recherche locale. Cet algorithme offre l'avantage d'être optimal et rapide dans la recherche globale, car ce dernier ne s'arrête pas lorsqu'il trouve un problème, mais il va essayer de chercher encore et encore jusqu'à ce qu'il trouve le problème le plus important. Cette procédure lui permet de s'échapper aux minima locaux. Cependant, pour

éviter une recherche infinie, nous avons limité la surface de recherche par un cercle pour chaque drone exécutant cet algorithme.

Algorithme 5 : Eagle Search Strategy

Result : Récupérer la dernière position du drone
Pr $\leftarrow NULL$
Initialisation des positions des drones(aléatoire)
while *(problème* $\neq$ *vrai) et (énergie* $>$ *seuil)* **do**
  Lancer la recherche global avec "Levy flights"
  Pr $\leftarrow$ *problme*
end
while *problème = vrai) et (énergie* $>$ *seille)* **do**
  Lancer la recherche locale avec "fireflies algortihm"
  if *(new Prb* $>$ *pr)* **then**
    Pr $\leftarrow$ *newPrb*
  end
end

Recherche globale (Lèvy flights) : Grâce à cette méthode, on peut faire une recherche globale plus efficace par rapport aux autres méthodes de recherche. Au début le drone va chercher l'existence de problèmes dans une zone limitée précédemment ; pour cela on applique l'algorithme de "levy flights" en se basant uniquement sur le champ de vision et l'énergie du drone. Ce dernier utilise des déplacements aléatoires pour la recherche (Algorithme 5). Dès qu'un drone capture un ou plusieurs problèmes, il va changer son état vers la recherche locale.

Algorithme 6 : Levy Flights

Result : Empty
initialiser la position de drone;
while *(problème = false) and (énergie* $>$ *seuil)* **do**
  jump$\leftarrow objectiffunction(jumplength, angle)$
  jump randomly(jump)
end

Recherche locale (Fireflies) : L'exécution de cet algorithme est conditionnée par la détection préalable d'un problème au niveau de la zone de recherche. "Fireflies" permet de scanner les zones les plus proches du problème initial afin de vérifier l'existence d'un autre problème plus grand que le premier, si la condition est vérifiée le drone se déplace vers lui et lance une autre recherche jusqu'à ce qu'il trouve le problème le plus grand.

Après la détection du problème dans l'étape précédente, le drone va utiliser son champ de vision pour détecter s'il existe un problème plus grave. Si un tel problème existe, le drone se déplace vers celui-ci. De la même manière, il essaye de détecter l'existence d'un autre problème plus grave. Le drone se stabilise lorsque le problème en cours de traitement est le plus grand dans son champ de vision.

Algorithme 7 : Fireflies

```
Result : Récupérer la dernière position du drone
fonction objective f(x),x=(x1...xd)
Initialiser la population de drones xi(i=1...n)
Le degré du problème Ii dans xi est déterminé par f(xi)
while (énergie > seuil) et (plus grave problème non trouvé) do
    for i ← 1 to n do
        for j ← 1 to i do
            if (Ij>Ii) then
                Déplacer le drone de i vers j
            end
            Trouver d'autres problèmes dans le champs de vision et
             mise à jour
        end
    end
end
```

3.3.6 Phase de coordination et de communication (GWO)

Dans cette phase, le drone qui a détecté le problème le plus grand va envoyer un message pour demander de l'aide aux autres drones qui sont proches du problème en cours de traitement. Les drones qui se trouvent à proximité du problème et qui n'ont pas de tache critique vont répondre par un message positif et se déplacent vers le drone qui a envoyé le message. La coordination et la communication entre les drones sont gérées par la stratégie des loups gris (Grey Wolves). Pour cela, on a besoin d'un chef (alpha) qui a la meilleure vision ; les autres drones encerclent le problème ensuite, se rapprochent et changent leurs positions par rapport à la position d'alpha,

bêta et delta. Le champ de vision est calculé à chaque fois pour essayer de couvrir le problème à 80% au minimum en utilisant les drones nécessaires. En même temps, si un drone actif a consommé son énergie, il sera remplacé par un autre drone ayant la capacité de continuer le processus.

Algorithme 8 : Grey Wolf Optimazer

Result : Xa,Xb,Xd

Initialiser a,A,C

Calculer le champs de vision de chaque drone

Xa=le drone qui a le meilleur champ de vision

Xb=le drone qui a le deuxième meilleur champ de vision

Xd=le drone qui a le troisième meilleur champs de vision

while *(énergie > seuil) et (problème = true)* **do**

  for *chaque drone* **do**

    Mise à jour de la position actuelle avec l'équation :

    X(t+1)=(X1+X2+X3)/3

  end

  Mise à jour de a,A,C.

  Calculer le champ de vision de chaque drone

  Mise à jour de Xa,Xb,Xd

end

3.3.7 Phase de remplacement

Dans cette phase les drones ayant une batterie faible (inférieur à un seuil) seront automatiquement changés par d'autres drones ayant des batteries pleines que ce soit dans la même équipe ou en utilisant un autre drone libre. Pour cela, on va appliquer l'algorithme suivant :

Algorithme 9 : Algorithme de remplacement

```
if (droneX.batterie <= seuil) then
    droneX envoie un message au drone.Chef contenant son état de
     batterie
    drone.chef envoie un message à l'équipe pour trouver un autre
     drone pour le remplacement avec la plus grande batterie.
    if il y a un drone.remplaçant dans l'équipe then
        drone.chef envoie un message au drone.remplaçant contenant
         la position du droneX
        drone.remplaçant prend la position du droneX
        droneX va vers le station pour recharger
    else
        if (il y a un drone.libre dans la zone) then
            drone.chef envoie un message au drone.libre contenant la
             position du droneX
            drone.libre prend la position du droneX
            droneX va vers le station pour recharger
        if (il y a pas un drone.libre) then
            droneX va vers le station pour charger et revenir
    end
end
```

au niveau simulation, l'énergie du drone est modélisé en temps de traitement et la taille du problème par le nombre de pixels qui le représente, par exemple :

— Un drone avec 100% d'énergie est modélisé par 10 minutes ; donc chaque 6 secondes, l'énergie diminue de 1%.

— Un problème d'une taille 6px nécessite 2 minutes pour le résoudre et

un autre problème d'une taille 3px nécessite juste 1 minute pour le résoudre.

3.4 Conclusion

Nous avons présenté dans ce chapitre l'architecture principale de notre système ainsi que les différents modules qui le compose. Nous avons divisé le problème de détection et de suivi en sous-problèmes et nous avons choisi l'algorithme le plus adéquat pour chacun d'eux. Tous les algorithmes proposés ont été légèrement modifiés afin d'être conformes le plus possible avec nos objectifs. L'utilisation de l'algorithme de l'aigle pour la détection des problèmes avec la recherche locale et la recherche globale a permis aux drones de bouger librement afin de trouver le problème le plus grand. Cette démarche a permis de visiter tous les cas possibles et de sélectionner celui qui a le plus d'importance en termes d'urgence (cas d'accident). L'algorithme de Grey Wolves quant à lui a permis de coordonner et de guider l'opération de surveillance par le minimum de drones et le maximum de vision.

Nous allons dans le prochain chapitre présenter les détails relatifs à l'implémentation et montrer quelques résultats de la détection et du traitement des problèmes dans des scénarios aléatoires.

CHAPITRE 4
IMPLÉMENTATION & RÉSULTATS

4.1 Introduction

Les drones sont des robots fréquemment utilisés dans diverses missions telles que la surveillance des zones dangereuses, les secours, le suivi des objets, etc. Comme nous l'avons mentionné dans la section précédente, le groupe de drones peut avoir besoin de travailler ensemble via l'échange d'informations sur leur état et leur environnement dans le but d'accomplir la mission prévue avec un minimum d'effort et un maximum de qualité.

Dans ce chapitre, nous allons valider la proposition présentée dans le chapitre précédent. L'absence d'un simulateur public qui offre une certaine flexibilité en termes de communication et d'autonomie nous a conduits à développer notre propre simulateur. Nous commençons par la présentation de l'environnement matériel et logiciel utilisé pour le développement de notre simulateur, ensuite, nous présentons notre modélisation et enfin, nous terminerons par la présentation de notre simulateur.

4.2 Environnement de réalisation

4.2.1 Environnement matériel

Le développement de notre application est réalisé par ordinateur avec les caractéristiques suivantes :

- ✓ **Système d'Exploitation (SE) :** Linux Ubuntu 18.04.5 LTS 64-bit
- ✓ **Processeur :** Intel® Core™ i5-3340 CPU @ 3.10GHz 4
- ✓ **Disque dur :** 486,3 GB
- ✓ **Mémoire vive (RAM) :** 3,7 GiB
- ✓ **Carte graphique (ou Carte vidéo) :** Intel® HD Graphics 2500 (IVB GT1)

4.2.2 Environnement Logiciel

Python : Python est un langage de programmation de haut niveau interprété, orienté objet, très attractif pour le développement rapide d'applications, ainsi que pour une utilisation comme langage de script ou de collage pour connecter des composants existants entre eux. La syntaxe de Python est simple et facile à apprendre. Elle met l'accent sur la lisibilité et réduit donc le coût de la maintenance. Python prend en charge les modules et les packages, ce qui encourage la modularité du programme et la réutilisation du code [84]. Dans notre travail, nous avons utilisé **la version 3.6.9**.

Python est un langage qui supporte plusieurs paradigmes de programmation. Parmi ces paradigmes, le paradigme Multithreading que nous avons utilisé.

Le Paradigme Multithreading : On a utilisé aussi **les threads** pour être capable de construire un système distribuer. Un thread est similaire à un processus, car tous deux représentent l'exécution d'un ensemble d'instructions du langage machine d'un processeur, du point de vue de l'utilisateur. Ces exécutions semblent se dérouler en parallèle [77]. Les threads ont trois caractéristiques principales [78].

La première caractéristique concerne le calcul distribué ou le calcul parallèle. Grâce à cette caractéristique, l'exécution des tâches indépendantes sera donc plus rapide, car l'exécution peut être effectuée simultanément.

La deuxième caractéristique est le raccordement de threads avec des interfaces graphiques. Cette caractéristique a permis de développer des interfaces graphiques contrôlées indépendamment par un ou un ensemble de threads grâce à la propriété précédente. L'importance de cette caractéristique se manifeste dans le développement des jeux.

La troisième caractéristique concerne la communication entre ordinateurs ou plus généralement la communication Internet. Avec cette caractéristique, il est possible de développer un système totalement distribué. Cette caractéristique permet au système de ne pas rester bloqué dans l'état d'attente des messages, mais il est capable d'écouter les messages qu'il reçoit via un port par un thread, et permet à d'autres threads d'effectuer d'autres tâches indépendamment de la tâche de l'autre thread.

Notre contexte de travail nécessite une interface graphique. Python a plusieurs modules graphiques. Dans notre implémentation, nous utilisons le module **Turtle graphics** [**turtle**].

Turtle graphics : Turtle graphics ou les graphiques de tortues sont un moyen populaire pour le développement graphique des jeux. Il faisait partie du langage de programmation Logo original développé par Wally Feurzeig et Seymour Papert en 1966.

Le module Turtle est un module intégré en python qui ne nécessite pas une installation supplémentaire. Turtle est un stylo imaginaire utilisé pour créer des formes et des dessins 2D par des fonctions de contrôle (telles que : avancer, tourner, orienter, poser le stylo, relever le stylo, couleur et largeur du stylo, et ainsi de suite). Les principaux avantages de ce module est qu'il est extrêmement simple et permet de dessiner très facilement des choses à l'écran. Cependant, le module Turtle prend en charge certaines fonctionnalités avancées dont nous aurons besoin dans la programmation.

Le module tortue fournit des primitives graphiques de tortue, à la fois de manière orientée objet et orientée procédure. Comme il utilise la bibliothèque Tkinter pour les graphiques sous-jacents.

PyCharm : PyCharm est un environnement de développement intégré (IDE) pour le langage de programmation Python développé par la société tchèque JetBrains. PyCharm est un éditeur multi-plafrome qui fonctionne sous Windows, Mac et Linux [64]. Dans notre travail on a utilisé **la version** 2020.2.2.

4.3 Présentation du Simulateur

4.3.1 Architecture Logicielle

La Figure 4.1 schématise l'architecture logicielle de notre simulateur développé en Python basé sur le paradigme «Multithreading» et la bibliothèque

graphique «Turtle graphics» présentés dans la section précédente (Section 4.2.2). Comme le montre la Figure 4.1, notre simulateur se compose de trois modules principaux : les drones, le Messenger et les problèmes, en plus du protocole de communication.

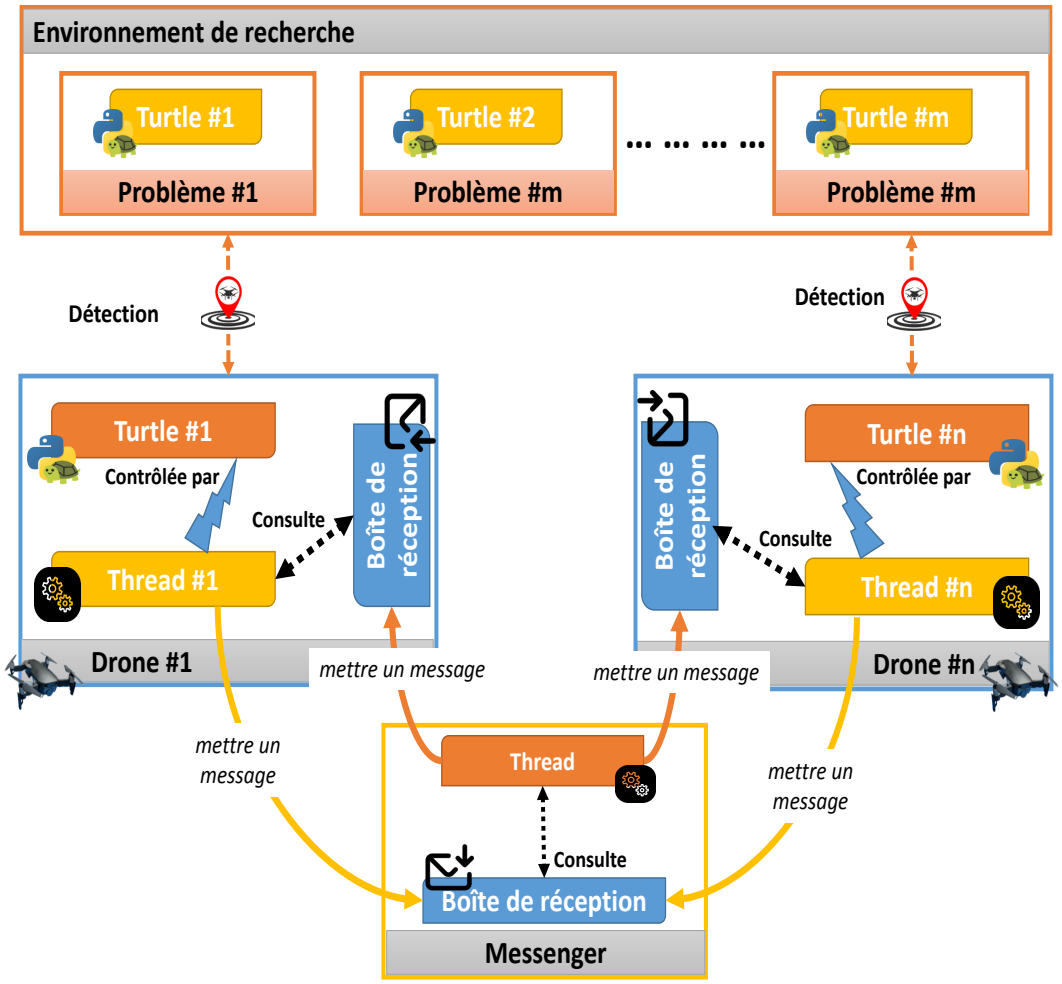


FIGURE 4.1 – Architecture logicielle du simulateur

Les drones : Les drones sont des entités autonomes modélisées par :

- *Un Turtle* : pour la représentation graphique du drone
- *Un Thread* : représente le cerveau du drone, c'est lui qui contrôle les mouvements du Turtle correspondant, échange des données avec d'autres

drones et qui fait les calculs qui font que le drone interagit avec les autres drones et avec l'environnement (les problèmes comme les accidents) selon les algorithmes présentés dans le chapitre 2. En d'autres termes, le Thread désigne le comportement du drone.

Chaque drone a un champ de vision modélisé par un cercle 2D. Selon notre conception, le champ de vision du drone est composé de deux parties : champ de vision local et champ de vision global comme le montre la Figure 4.2.

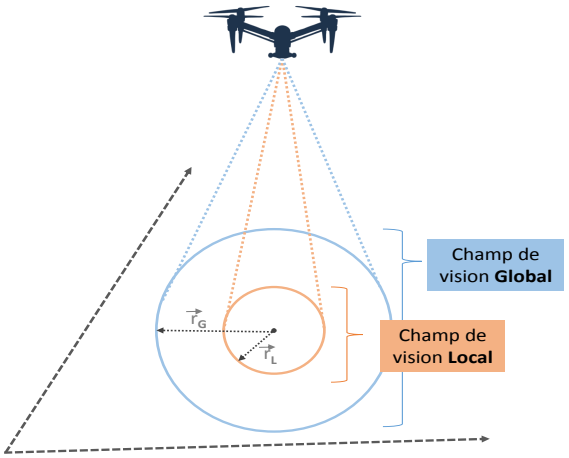


FIGURE 4.2 – Modélisation du champ de vision du drone sur un plan 2D

Le Messenger : Le Messenger est un élément principal dans notre simulateur. C'est lui qui contrôle l'échange de données entre les drones pendant la mission selon un simple protocole. Pour simplifier, on peut imaginer ce Messenger comme un "facteur" qui permet de transmettre un message du drone émetteur vers le drone récepteur.

L'environnement : L'environnement est l'espace autorisé dans lequel les drones effectuent leurs recherches. Ce dernier contient un ensemble de problèmes qui sont modélisés par des objets Turtles.

Les Problèmes : Les problèmes dans notre contexte sont modélisés par des objets Turtles et représentés graphiquement par des cercles distribués aléatoirement dans l'environnement. Chaque problème a un diamètre représentant le degré du problème (voire la Figure 4.3).

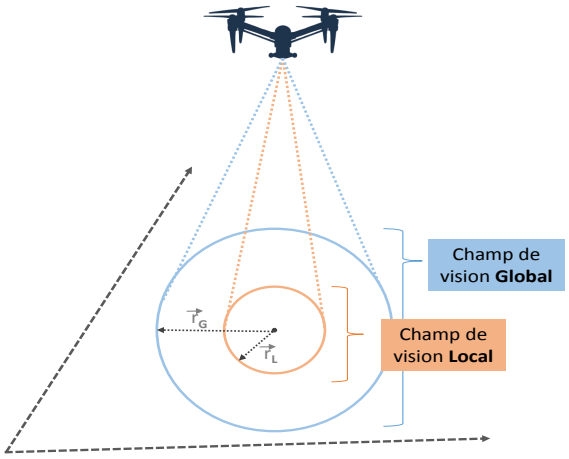


FIGURE 4.3 – Modélisation de la zone du problème sur un plan 2D

Le protocole de communication : Pour une communication fiable entre les drones lors de la mission, nous proposons un protocole de communication très simple basé sur le principe de la facture et de la boîte aux lettres utilisée dans la réalité. Ce protocole est composé de trois acteurs : l'émetteur, le récepteur et le facteur (ou Messenger). Chacun de ces acteurs a une boîte aux lettres qui contient les messages reçus.

Le principe de ce protocole est simple. Si un drone a besoin de transmettre un message à un autre drone ou à un ensemble de drones, iI suffit d'envoyer le message au Messenger (mettre le message dans sa boîte aux lettres). Le Messager reste toujours en écoute pendant le déroulement de la simulation.

Si le messager trouve qu'il y a un message dans sa boîte aux lettres, il retransmet à son tour le message au(x) drone(s) concerné(s). Afin de simplifier ce protocole de communication, nous avons identifié trois types de messages :

- ✉ *Message simple* : ce type de message est utilisé lorsqu'un drone a besoin de communiquer avec un autre drone bien spécifique.
- ✉ *Message de diffusion (broadcast)* : il y a des cas où un drone doit transmettre des informations, des ordres ou des demandes à tous les drones du système. Dans ce cas, le drone envoie un message de diffusion. Le Messenger comprend qu'il est destiné à tous les drones.
- ✉ *Message Sub-diffusion* : la sub-diffusion est un cas particulier de la diffusion. Ce type est utilisé dans le cas où un drone a besoin de communiquer avec un ensemble de drones spécifiques pour transmettre une information, une commande ou une demande.

La Figure 4.4 montre en détail la composition de ces types de messages.

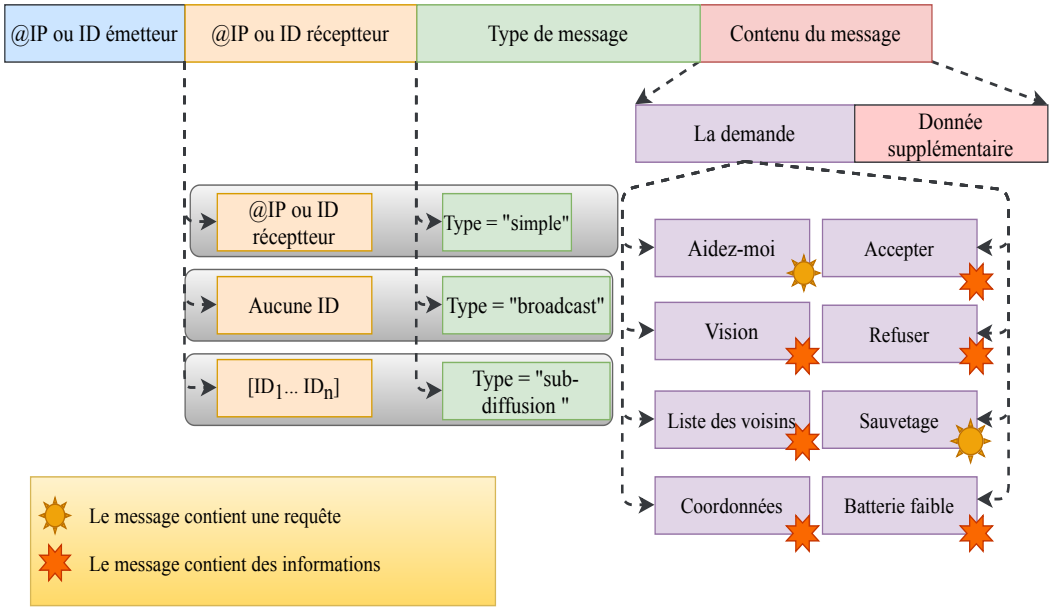


FIGURE 4.4 – Composition des messages.

4.4 Implémentation

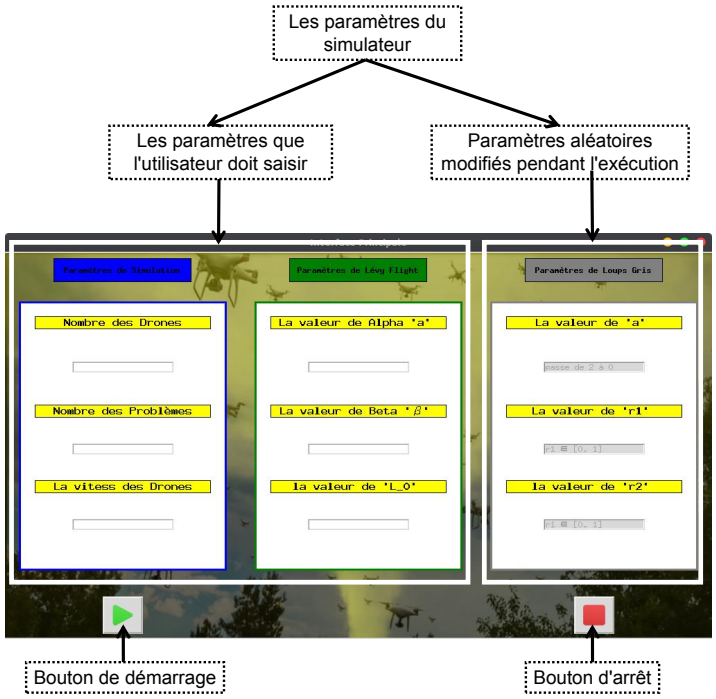


FIGURE 4.5 – Interface principale.

Dans cette partie, nous allons présenter notre simulateur. L'interface graphique principale du simulateur comprend un ensemble de composants à partir desquels on peut facilement paramétrer le système. La figure 4.5 montre cette interface, qui s'affiche au démarrage de l'exécution. Dans cette interface, il y a trois groupes de paramètres : le premier (en bleu) contient les paramètres initiaux du simulateur (nombre de drones, nombre de problèmes et vitesse du drone). Le deuxième groupe (en vert) contient les paramètres de l'algorithme "Lèvy Flight" et le dernier groupe (en gris) contient tous des paramètres prédéfinis de l'algorithme du loup gris (GWO). L'interface contient également le bouton de démarrage et d'arrêt de la simulation.

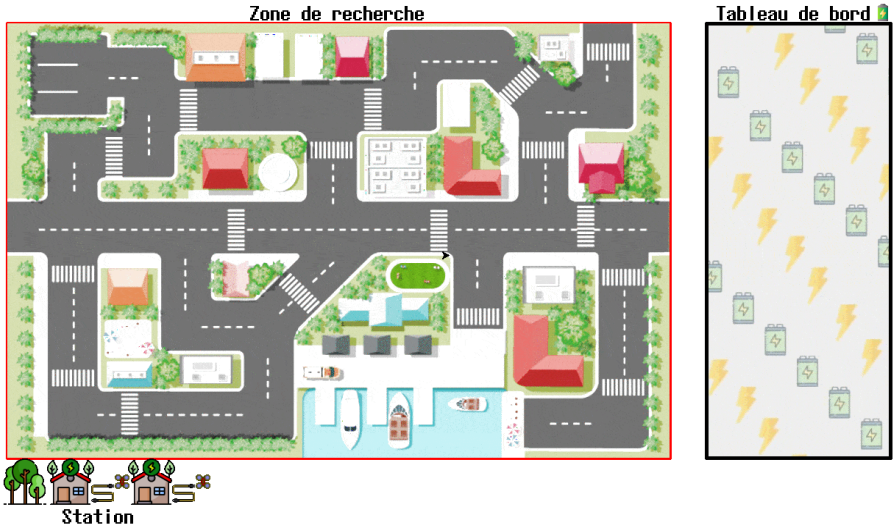


FIGURE 4.6 – Vue globale de notre simulateur.

La Figure 4.6 montre un aperçu de notre simulateur qui se compose de trois composants principaux : *La zone de recherche*, qui modélise l'espace autorisé dans lequel les drones effectuent leurs recherches, *le tableau de bord* qui affiche l'état actuel de la charge des drones et *la station de base* qui permet de changer ou recharger les batteries des drones.

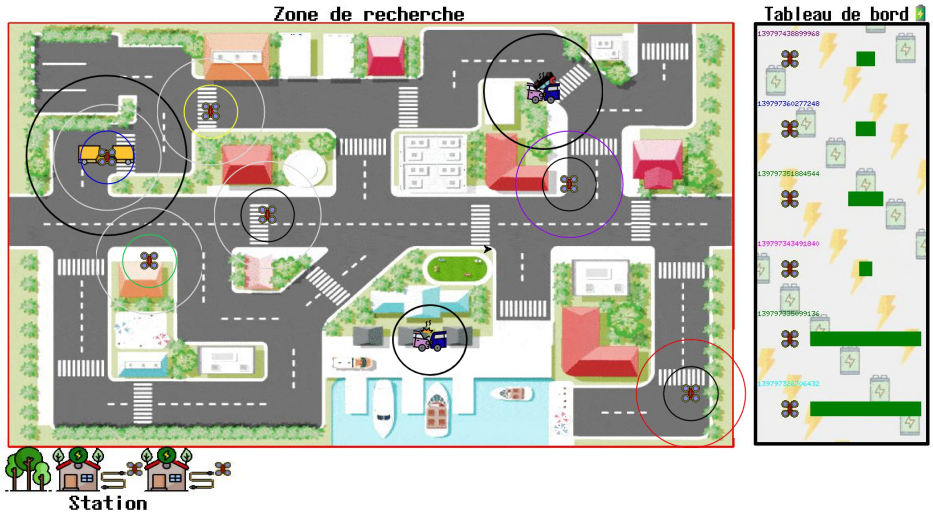


FIGURE 4.7 – Scénario général.

La figure 4.7 présente un scénario général qui détermine les différentes étapes du drone pendant une mission et la figure 4.8 indique la signification des couleurs du drone (cercles) et des problèmes.

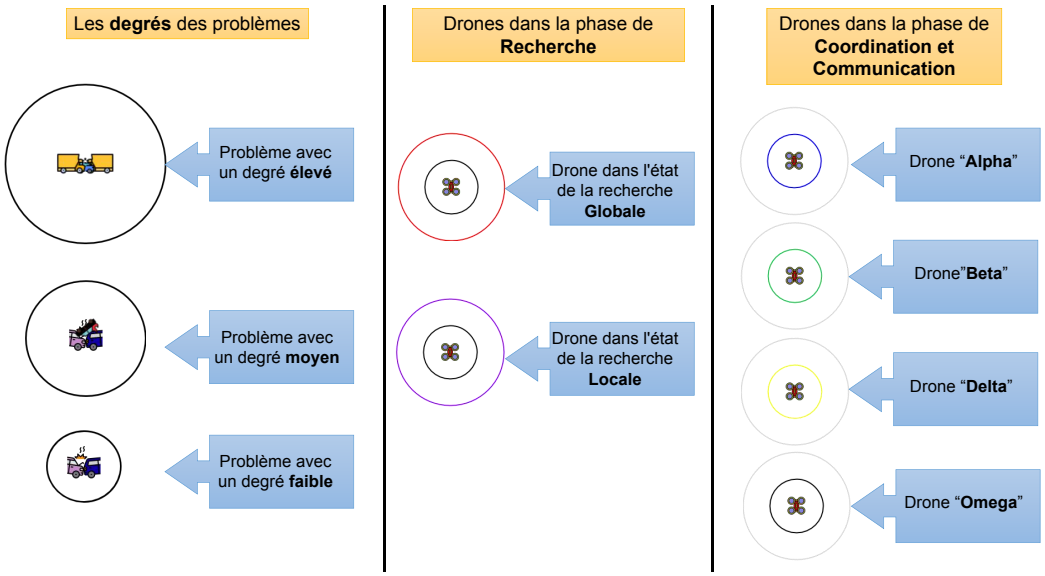


FIGURE 4.8 – Scénario général.

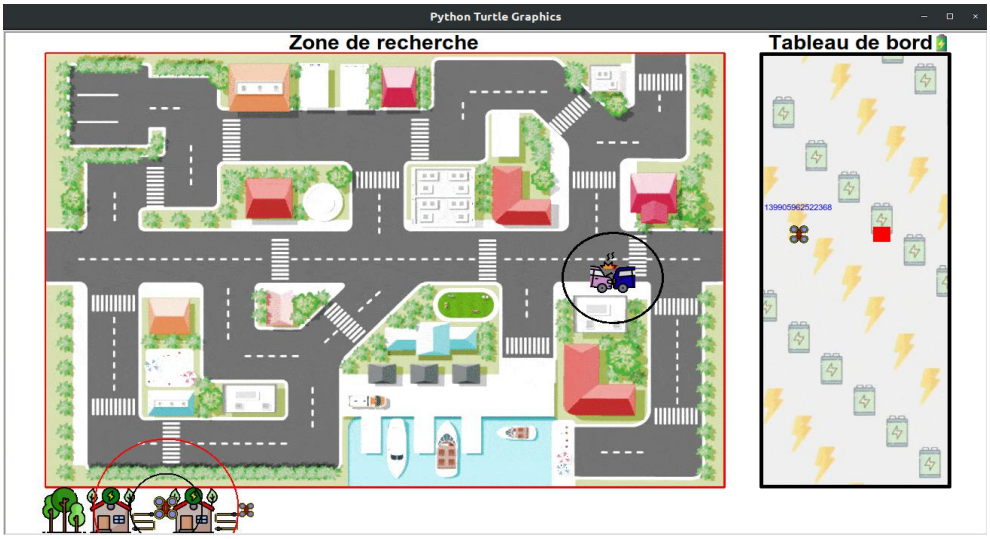


FIGURE 4.9 – Drone exclu.

Les Figures 4.9 et 4.10 montrent respectivement le cas d'un drone qui a été exclu de la mission en raison de sa faible énergie et un nouveau drone sort de la station pour remplacer le drone exclu. Le nouveau drone conserve la même identification et le même état que le drone remplacé pour poursuivre la mission à partir du point de départ du drone exclu.

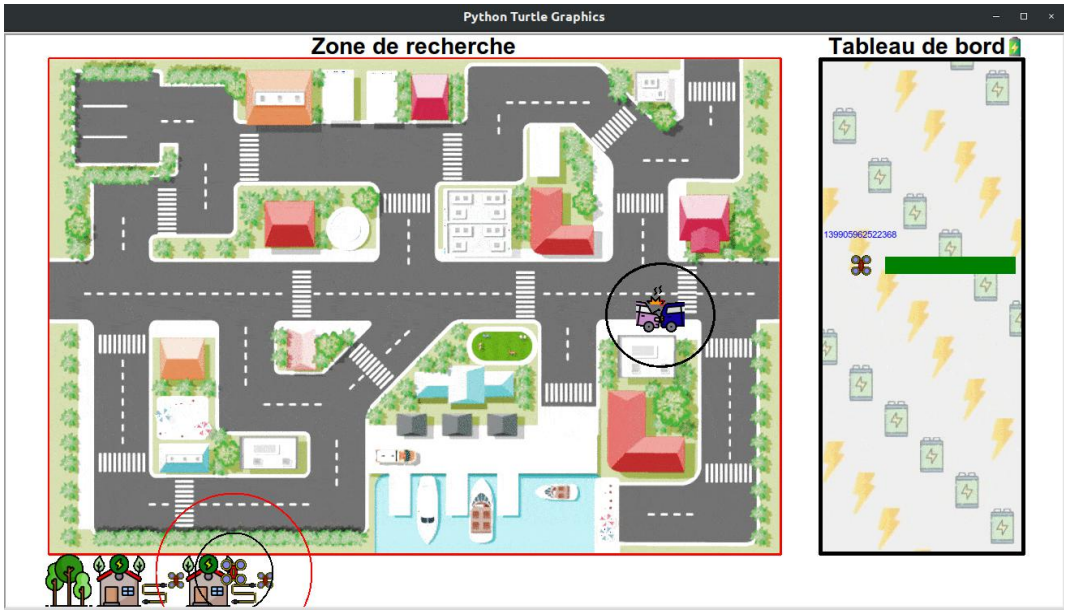


FIGURE 4.10 – Drone remplaçant.

4.5 Conclusion

Dans ce chapitre, nous avons présenté en détail notre simulateur développé en deux parties : l'architecture logicielle et l'implémentation. Dans la première partie, nous avons expliqué l'architecture logicielle qui caractérise le simulateur. Dans la partie d'implémentation, nous avons présenté l'environnement et les scénarios de simulation.

CONCLUSION GÉNÉRALE

Dans ce mémoire de fin d'études, nous avons présenté un système coopératif et distribué basé sur les drones de surveillance dans le but d'offrir une bonne détection des problèmes et une meilleure gestion des interventions. Le système proposé permet de surmonter les difficultés rencontrées par les approches à base d'un seul drone et qui est, dans la plupart des cas, insuffisantes pour fournir des tâches opérationnelles complètes, en raison de leur limitation dans la capacité de traitement et la flexibilité dans l'exécution des tâches conditionnées par le temps de vol (énergie).

Pour aboutir à de meilleures performances, notre système a été divisé en trois modules. Chaque module implémente un algorithme métaheuristique qui permet de gérer au mieux les taches désirées. La première phase consiste à chercher dans la zone de surveillance l'existence d'un éventuel problème à résoudre. Cette dernière s'effectue grâce à l'algorithme de l'aigle en deux étapes, à savoir la recherche globale et la recherche locale. Dans la recherche globale, le champ de vision est très large permettant au drone de surveillance de détecter des anomalies dans la zone. Cette dernière est assurée par l'algorithme "Levy Flights" qui permet un déplacement optimal au drone dans la

zone de surveillance. La détection d'un problème déclenche automatiquement la recherche locale qui est assurée par l'algorithme "Fireflies". Ce dernier gère le déplacement du drone à proximité du problème principal afin de trouver d'autres problèmes plus prioritaires et qui nécessitent une intervention immédiate. C'est à ce moment-là que le système passe d'une surveillance basée sur un seul drone vers une surveillance multi drone. Ce passage active la deuxième phase qui consiste à coopérer et faire communiquer les drones par le biais de l'algorithme des "Grey Wolves" pour maximiser la performance. Pour minimiser le risque de pannes dû à la diminution des batteries, le dernier module assure une stratégie de remplacement optimale des drones.

Afin de tester les algorithmes proposés, nous avons implémenté notre propre simulateur qui permet la création aléatoire de plusieurs problèmes dans la surface de surveillance et d'assurer la résolution par un essaim de drones. Les résultats de la simulation sont très encourageants, le système a réussi à gérer de façon optimale tous les problèmes générés dans différents scénarios.

Comme perspectives, nous proposons de tester d'autres algorithmes méthaheuritiques et d'implémenter le système dans un environnement distribué.

BIBLIOGRAPHIE

[1] Rana ABDALLAH. « Reliability approaches in networked systems : Application on Unmanned Aerial Vehicles ». Thèse de doct. 2019.

[2] *AVANTAGES DU DRONE*. `http://ufly-drones.com/avantages-du-drone/`. (date d'accès : 21/09/2020).

[3] Ema Falomir BAGDASSARIAN. « Calcul dynamique de chemin par des drones autonomes en essaim compact dans le cadre de missions en environnements complexes ». Thèse de doct. Université de Bordeaux, 2019.

[4] Malti BAGHEL, Shikha AGRAWAL et Sanjay SILAKARI. « Survey of metaheuristic algorithms for combinatorial optimization ». In : *International Journal of Computer Applications* 58.19 (2012).

[5] Wolfgang BANZHAF et al. *Genetic programming*. Springer, 1998.

[6] Ilker BEKMEZCI, Ozgur Koray SAHINGOZ et Şamil TEMEL. « Flying ad-hoc networks (FANETs) : A survey ». In : *Ad Hoc Networks* 11.3 (2013), p. 1254-1270.

[7] Gerardo BENI et Jing WANG. « Swarm intelligence in cellular robotic systems ». In : *Robots and biological systems : towards a new bionics ?* Springer, 1993, p. 703-712.

[8] Eric BONABEAU et Christopher MEYER. « Swarm intelligence : A whole new way to think about business ». In : *Harvard business review* 79.5 (2001), p. 106-115.

[9] G BRAMMYA et al. « Deer hunting optimization algorithm : A new nature-inspired meta-heuristic paradigm ». In : *The Computer Journal* (2019).

[10] Shi-Zhong BU et al. « Progesterone induces apoptosis and up-regulation of p53 expression in human ovarian carcinoma cell lines ». In : *Cancer : Interdisciplinary International Journal of the American Cancer Society* 79.10 (1997), p. 1944-1950.

[11] Ann CAVOUKIAN. *Privacy and drones : Unmanned aerial vehicles*. Information et Privacy Commissioner of Ontario, Canada Ontario, 2012.

[12] *c'est quoi un drone*. `http://www.afcadillac.net/_serveurs/drone/questce_quun_drone_.html`. (date d'accès : 08/09/2020).

[13] Min-Yuan CHENG et Doddy PRAYOGO. « Symbiotic organisms search : a new metaheuristic optimization algorithm ». In : *Computers & Structures* 139 (2014), p. 98-112.

[14] João Pedro Marques COSTA. « Drone flock : formation control of multi-drones ». In : (2019).

[15] Erik CUEVAS et al. « A swarm optimization algorithm inspired in the behavior of the social-spider ». In : *Expert Systems with Applications* 40.16 (2013), p. 6374-6384.

[16] ML CUMMINGS. « Operator interaction with centralized versus decentralized UAV architectures ». In : *Handbook of Unmanned Aerial Vehicles* (2015), p. 977-992.

[17] Swagatam DAS et al. « Bacterial foraging optimization algorithm : theoretical foundations, analysis, and applications ». In : *Foundations of computational intelligence volume 3*. Springer, 2009, p. 23-55.

[18] Dipankar DASGUPTA et Stephanie FORREST. « Artificial immune systems in industrial applications ». In : *Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials. IPMM'99 (Cat. No. 99EX296)*. T. 1. IEEE. 1999, p. 257-267.

[19] Lawrence DAVIS. « Handbook of genetic algorithms ». In : (1991).

[20] *Diviser pour régner*. `https://pixees.fr/informatiquelycee/n_site/nsi_term_algo_diviser_pour_regner.html`. (date d'accès : 08/09/2020).

[21] Tansel DOKEROGLU et al. « A survey on new generation metaheuristic algorithms ». In : *Computers & Industrial Engineering* 137 (2019), p. 106040.

[22] Marco DORIGO, Mauro BIRATTARI et Thomas STUTZLE. « Ant colony optimization ». In : *IEEE computational intelligence magazine* 1.4 (2006), p. 28-39.

[23] Adrien DROUOT. « Stratégies de commande pour la navigation autonome d'un drone projectile miniature ». Thèse de doct. 2013.

[24] Russell C EBERHART, Yuhui SHI et James KENNEDY. *Swarm intelligence*. Elsevier, 2001.

[25] *firstdrone*. `http://lhistoireenrafale.lunion.fr/2017/07/02/2-juillet-1917-capitaine-boucher-laeroplane-pilote/`. (date d'accès : 06/10/2020).

[26] Pravin S GAME, Dr VAZE et al. « Bio-inspired Optimization : metaheuristic algorithms for optimization ». In : *arXiv preprint arXiv :2003.11637* (2020).

[27] Amir Hossein GANDOMI et Amir Hossein ALAVI. « Krill herd : a new bio-inspired optimization algorithm ». In : *Communications in nonlinear science and numerical simulation* 17.12 (2012), p. 4831-4845.

[28] Zong Woo GEEM, Joong Hoon KIM et Gobichettipalayam Vasudevan LOGANATHAN. « A new heuristic optimization algorithm : harmony search ». In : *simulation* 76.2 (2001), p. 60-68.

[29] Fred GLOVER et Manuel LAGUNA. « Tabu search ». In : *Handbook of combinatorial optimization*. Springer, 1998, p. 2093-2229.

[30] David E GOLDBERG. « Genetic algorithms in search ». In : *Optimization, and MachineLearning* (1989).

[31] Jin-Kao HAO, Philippe GALINIER et Michel HABIB. « Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes ». In : *Revue d'intelligence artificielle* 13.2 (1999), p. 283-324.

[32] Boukef Ben Otman HELA. « Sur l'ordonnancement d'ateliers job-shop flexibles et flow-shop en industries pharmaceutiques Optimisation par algorithmes génétiques et essaims particulaires ». Thèse de doct. École centrale de Lille & Université de Tunis El-manar, École nationale d'ingénieurs de Tunis, 2013.

[33] *Influence of Environmental Conditions on the Huddling Behavior of Emperor Penguins*. `https://bio.physik.fau.de/2017/11/14/influence-of-environmental-conditions-on-the-huddling-behavior-of-emperor-penguins/`. (date d'accès : 04/05/2020).

[34] Vincent JANICOT. « Simulation des circuits électroniques RF/Analogiques/Numériques excités par des signaux à modulation complexe ». Thèse de doct. Université Joseph-Fourier-Grenoble I, 2002.

[35] Raf JANS et Zeger DEGRAEVE. « Meta-heuristics for dynamic lot sizing : A review and comparison of solution approaches ». In : *European journal of operational research* 177.3 (2007), p. 1855-1875.

[36] Anis Farhan KAMARUZAMAN et al. « Levy flight algorithm for optimization problems-a literature review ». In : *Applied Mechanics and Materials*. T. 421. Trans Tech Publ. 2013, p. 496-501.

[37] Dervis KARABOGA. *An idea based on honey bee swarm for numerical optimization*. Rapp. tech. Technical report-tr06, Erciyes university, engineering faculty, computer . . ., 2005.

[38] Dervis KARABOGA et Bahriye BASTURK. « A powerful and efficient algorithm for numerical function optimization : artificial bee colony (ABC) algorithm ». In : *Journal of global optimization* 39.3 (2007), p. 459-471.

[39] Soheyl KHALILPOURAZARI et Saman KHALILPOURAZARY. « A Robust Stochastic Fractal Search approach for optimization of the surface grinding process ». In : *Swarm and Evolutionary Computation* 38 (2018), p. 173-186.

[40] Soheyl KHALILPOURAZARI et Alireza Arshadi KHAMSEH. « Bi-objective emergency blood supply chain network design in earthquake considering earthquake magnitude : a comprehensive study with real world application ». In : *Annals of Operations Research* 283.1 (2019), p. 355-393.

[41] Mouhyemen KHAN et al. « Mobile target coverage and tracking on drone-be-gone UAV cyber-physical testbed ». In : *IEEE Systems Journal* 12.4 (2017), p. 3485-3496.

[42] Marek KULBACKI et al. « Survey of drones for agriculture automation from planting to harvest ». In : *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*. IEEE. 2018, p. 000353-000358.

[43] Anuj KUMAR, Sangeeta PANT et Mangey RAM. « System reliability optimization using gray wolf optimizer algorithm ». In : *Quality and Reliability Engineering International* 33.7 (2017), p. 1327-1335.

[44] Farid LACHOURI et al. « Proposition d'un nouvel algorithme d'optimisation basé sur la théorie de la gravitation ». Thèse de doct. Juil. 2018, p. 80. DOI : 10.13140/RG.2.2.12118.40009.

[45] Alina LAZAR. « Heuristic knowledge discovery for archaeological data using genetic algorithms and rough sets ». In : *Heuristic and Optimization for Knowledge Discovery*. IGI Global, 2002, p. 263-278.

[46] Chunbo LUO et al. « Unmanned aerial vehicles for disaster management ». In : *Geological Disaster Monitoring Based on Sensor Networks*. Springer, 2019, p. 83-107.

[47] *Lévy flights. Foraging in a finance blog*. https://quantdare.com/levy-flights/. (date d'accès : 09/04/2020).

[48] Emmanuel MA MUNG. « La dispersion comme ressource ». In : *Cultures & conflits* 33-34 (1999).

[49] João P MARQUES-SILVA et Karem A SAKALLAH. « GRASP : A search algorithm for propositional satisfiability ». In : *IEEE Transactions on Computers* 48.5 (1999), p. 506-521.

[50] Amel MENASRIA. « Etude comparative de la répartition optimale des puissances d'un réseau d'énergie électrique ». Thèse de doct. Université des Sciences et de la Technologie d'Oran, Mohamed Boudiaf, 2013.

[51] *Metaheuristics Network Website*. `http://www.metaheuristics.org/`. (date d'accès : 02/08/2020).

[52] Seyedali MIRJALILI. « Dragonfly algorithm : a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems ». In : *Neural Computing and Applications* 27.4 (2016), p. 1053-1073.

[53] Seyedali MIRJALILI et Andrew LEWIS. « The whale optimization algorithm ». In : *Advances in engineering software* 95 (2016), p. 51-67.

[54] Seyedali MIRJALILI, Seyed Mohammad MIRJALILI et Andrew LEWIS. « Grey wolf optimizer ». In : *Advances in engineering software* 69 (2014), p. 46-61.

[55] Nenad MLADENOVIĆ et Pierre HANSEN. « Variable neighborhood search ». In : *Computers & operations research* 24.11 (1997), p. 1097-1100.

[56] *Métaheuristique*. `https://fr.wikipedia.org/wiki/M%C3%A9taheuristique`. (date d'accès : 02/08/2020).

[57] *Métaheuristique*. `https://fr.wikipedia.org/wiki/Métaheuristiques#Avantages_et_inconvénients`. (date d'accès : 09/09/2020).

[58] Aníbal OLLERO et Iván MAZA. *Multiple heterogeneous unmanned aerial vehicles*. T. 37. Springer, 2007.

[59] Ibrahim H OSMAN et Gilbert LAPORTE. *Metaheuristics : A bibliography*. 1996.

[60] Wen-Tsao PAN. « A new fruit fly optimization algorithm : taking the financial distress model as an example ». In : *Knowledge-Based Systems* 26 (2012), p. 69-74.

[61] E PERRIN et al. « Optimisation globale par stratégie d'évolution ». In : *RAIRO-Operations Research* 31.2 (1997), p. 161-201.

[62] *Pilotes de drone, photographes, caméramans, techniciens...* `https://www.drone-malin.com/pages/en-savoir-plus/les-drones/c-est-quoi-un-drone.html`. (date d'accès : 08/09/2020).

[63] Kenneth PRICE, Rainer M STORN et Jouni A LAMPINEN. *Differential evolution : a practical approach to global optimization*. Springer Science & Business Media, 2006.

[64] *PyCharm*. `https://www.pcmag.com/encyclopedia/term/pycharm`. (date d'accès : 04/10/2020).

[65] Ramin RAJABIOUN. « Cuckoo optimization algorithm ». In : *Applied soft computing* 11.8 (2011), p. 5508-5518.

[66] R Venkata RAO, Vimal J SAVSANI et DP VAKHARIA. « Teaching–learning-based optimization : a novel method for constrained mechanical design optimization problems ». In : *Computer-Aided Design* 43.3 (2011), p. 303-315.

[67] Esmat RASHEDI, Hossein NEZAMABADI-POUR et Saeid SARYAZDI. « GSA : a gravitational search algorithm ». In : *Information sciences* 179.13 (2009), p. 2232-2248.

[68] Stuart RUSSELL et Peter NORVIG. « Artificial intelligence : a modern approach ». In : (2002).

[69] Shahrzad SAREMI, Seyedali MIRJALILI et Andrew LEWIS. « Grasshopper optimisation algorithm : theory and application ». In : *Advances in Engineering Software* 105 (2017), p. 30-47.

[70] Patrik SCHMUCK et Margarita CHLI. « Multi-uav collaborative monocular slam ». In : *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, p. 3863-3870.

[71] *Sciences Drone*. `https://www.futura-sciences.com/sciences/definitions/aeronautique-drone-6174/`. (date d'accès : 08/09/2020).

[72] Pete SEILER, Aniruddha PANT et Karl HEDRICK. « Analysis of bird formations ». In : *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.* T. 1. IEEE. 2002, p. 118-123.

[73] Dan SIMON. « Biogeography-based optimization ». In : *IEEE transactions on evolutionary computation* 12.6 (2008), p. 702-713.

[74] Thomas STÜTZLE. « Applying iterated local search to the permutation flow shop problem ». In : (1998).

[75] Thomas STÜTZLE. « Local search algorithms for combinatorial problems-analysis, algorithms and new applications ». In : *DISKI-Dissertationen zur Künstliken Intelligenz. In x, Sankt Augustin, Germany* (1999).

[76] TECHFERRY. *Swarm Intelligence*.

[77] *threads*. `https://fr.wikipedia.org/wiki/Thread_(informatique)`. (date d'accès : 04/10/2020).

[78] *threads*. `http://www.xavierdupre.fr/app/teachpyx/helpsphinx/c_parallelisation/thread.html`. (date d'accès : 04/10/2020).

[79] Jose TORRES-JIMÉNEZ et Juan PAVÓN. *Applications of metaheuristics in real-life problems*. 2014.

[80] *Tout ce que vous devez savoir sur les drones*. https://www.studiosport.fr/guides/drones/tout-savoir-sur-les-drones.html. (date d'accès : 06/09/2020).

[81] Peter JM VAN LAARHOVEN et Emile HL AARTS. « Simulated annealing ». In : *Simulated annealing : Theory and applications*. Springer, 1987, p. 7-15.

[82] Stefan VOSS et al. *Meta-heuristics : Advances and trends in local search paradigms for optimization*. Springer Science & Business Media, 2012.

[83] Yang WEI et Li QIQIANG. « Survey on Particle Swarm Optimization Algorithm [J] ». In : *Engineering Science* 5.5 (2004), p. 87-94.

[84] *What is Python ? Executive Summary*. https://www.python.org/doc/essays/blurb/. (date d'accès : 04/10/2020).

[85] Marino WIDMER. « Les métaheuristiques : des outils performants pour les problèmes industriels ». In : *3ème Conférence Francophone de MOdélisation et SIMulation MOSIM*. T. 1. 2001, p. 25-27.

[86] *wiki*. https://en.wikipedia.org/wiki/Unmanned_aerial_vehicle. (date d'accès : 08/09/2020).

[87] Alexander C WOODS et Hung M LA. « Dynamic target tracking and obstacle avoidance using a drone ». In : *International Symposium on Visual Computing*. Springer. 2015, p. 857-866.

[88] Y XIN-SHE. « An introduction with metaheuristic applications ». In : *Engineering Optimization* (2010).

[89] Xin-She YANG. « A new metaheuristic bat-inspired algorithm ». In : *Nature inspired cooperative strategies for optimization (NICSO 2010)*. Springer, 2010, p. 65-74.

[90] Xin-She YANG. *Engineering optimization : an introduction with metaheuristic applications*. John Wiley & Sons, 2010.

[91] Xin-She YANG. « Firefly algorithm, stochastic test functions and design optimisation ». In : *arXiv preprint arXiv :1003.1409* (2010).

[92] Xin-She YANG. *Nature-inspired metaheuristic algorithms*. Luniver press, 2010.

[93] Xin-She YANG. *Nature-inspired optimization algorithms*. Elsevier, 2014.

[94] Xin-She YANG et Suash DEB. « Cuckoo search via Lévy flights ». In : *2009 World congress on nature & biologically inspired computing (NaBIC)*. IEEE. 2009, p. 210-214.

[95] Xin-She YANG et Suash DEB. « Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization ». In : *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*. Springer, 2010, p. 101-111.

[96] Xin-She YANG et Suash DEB. « Two-stage eagle strategy with differential evolution ». In : *International Journal of Bio-Inspired Computation* 4.1 (2012), p. 1-5.

[97] Evşen YANMAZ et al. « Communication and coordination for drone networks ». In : *Ad hoc networks*. Springer, 2017, p. 79-91.

[98] Xin YAO. « Evolutionary artificial neural networks ». In : *International journal of neural systems* 4.03 (1993), p. 203-222.

[99] Yi ZHOU et al. « Multi-UAV-aided networks : Aerial-ground cooperative vehicular networking architecture ». In : *ieee vehicular technology magazine* 10.4 (2015), p. 36-44.

bibdrone.bib