الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي والبحث العلمي

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

جامعة 20 أوت 1955 سكيكدة

Faculté des Sciences

Département d'informatique

Mémoire de fin d'étude

En vue de l'obtention du diplôme de Master

Filière : Informatique

Spécialité : Réseaux et systèmes distribués

Thème

Système de navigation multi-robots inspiré des volées d'oiseaux

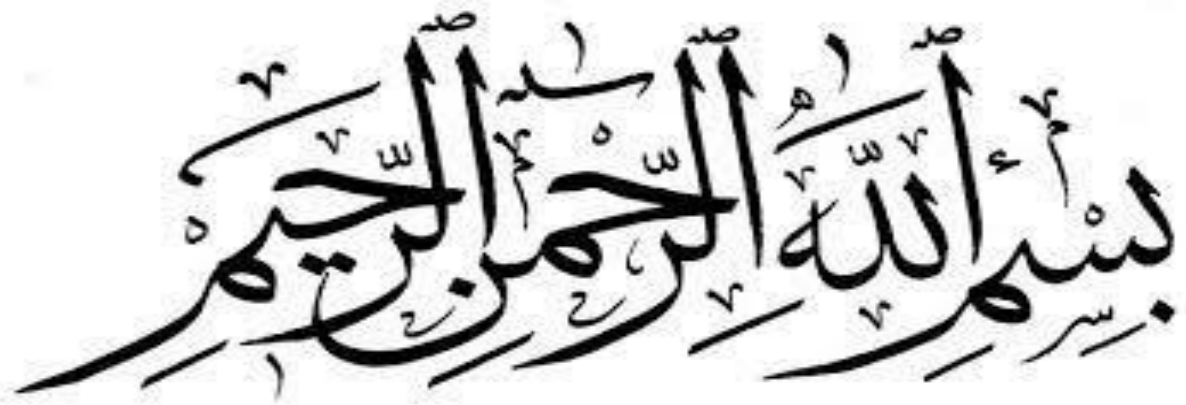
Présenté par :

- **REMMACHE Hala**
- **SAADHAMIDECHE Feyrouz**

Encadré par :

- **Mr BOUTINE Rachid**

Année universitaire :
2023/2024


بسم الله الرحمن الرحيم

Résumé

Le fil conducteur de notre projet était l'idée de se lancer dans une discipline révolutionnaire et émergeante pouvant apporter des solutions multiples où le champ d'intervention de l'être humain est plus ou moins réduit. En premier nous avons dérasé un état de l'art sur la robotique mobile, après, nous avons présenté quelques techniques de l'intelligence artificielle appliquées dans ce domaine, juste après, nous nous sommes concentrés sur les systèmes multi-robots et les techniques d'évitement d'obstacles d'inspiration biologique. Puis nous avons passé en revue les défis de coordination et de coopération entre les robots et nous avons expliqué les méthodes d'inspiration biologique telles que le modèle de Boid de Reynolds. En dernier, nous avons proposé un algorithme fusionnant (le champ de potentiel avec les règles de boids) qui répond à la problématique de navigation sans collision dans un espace inconnu.

Mots clés : champ de potentiel, robot, évitement d'obstacles, navigation autonome.

ملخص

كان القاسم المشترك لمشروعنا هو فكرة الانطلاق في الانضباط الثوري والناشئة التي يمكن أن تقدم حلولاً متعددة حيث مجال تدخل الكائن يتم تقليل الإنسان إلى حد ما. أولا قمنا بهدم حالة من الفن على الروبوتات المتنقلة، وبعد ذلك عرضنا بعض تقنيات الذكاء الاصطناعي تم تطبيقها في هذا المجال، وبعد ذلك مباشرة، ركزنا على أنظمة الروبوتات المتعددة وتقنيات تجنب العوائق المستوحاة بيولوجيًا. ثم استعرضنا تحديات التنسيق والتعاون بين الروبوتات وشرحنا الأساليب المستوحاة بيولوجيا مثل نموذج رينولدز بويد. أخيرًا، اقترحنا خوارزمية دمج (الحقل المحتمل مع قواعد بويد) والتي تستجيب لمشكلة التنقل الخالي من الاصطدام في مساحة غير معروفة.

الكلمات المفتاحية : المجال المحتمل , الروبوت, تجنب العوائق , التنقل المستقل.

Abstract

The common thread of our project was the idea of launching into a revolutionary discipline and emerging that can provide multiple solutions where the field of intervention of the being human is more or less reduced. First we demolished a state of the art on the mobile robotics, afterward, we presented some artificial intelligence techniques applied in this field, right after, we focused on multi-robot systems and biologically inspired obstacle avoidance techniques. Then we reviewed the coordination and cooperation challenges between robots and explained biologically inspired methods such as Reynolds' Boid model. Finally, we proposed an algorithm merging (the potential field with the boids rules) which responds to the problem of collision-free navigation in an unknown space.

Keywords: potential field, robot, obstacle avoidance, autonomous navigation.

Remerciement

Nous remercions, du plus profond de notre cœur, Dieu le tout puissant de nous avoir donné le courage et la volonté d'achever ce travail.

À nos chers parents, pour leur confiance, leurs encouragements et pour leurs sacrifices durant toute la vie, nous souhaitons que ce travail soit le fruit de leurs efforts…

Au Professeur BOUTINE Rachid, enseignant au département d'informatique, pour avoir accepté de diriger ce travail, en nous faisant bénéficier de son expérience, ses conseils et son aide.

Nous avons eu la chance et le plaisir d'effectuer ce travail de recherche. Nous tenons à vous exprimer notre profond respect et notre gratitude.

Aux membres du jury, pour le grand honneur qu'ils nous ont fait en acceptant de juger ce travail, nous souhaitons que cette dissertation soit le témoignage de notre reconnaissance et de notre profond respect.

Nous remercions aussi toutes les personnes qui ont contribué à nous transmettre le savoir scientifique durant toute la durée de nos études universitaires.

À tous ceux qui, de près ou de loin, ont contribué à la finalisation de ce travail et tous ceux qui ont souhaité nous voir arriver à ce stade.

Dédicace

A mon enseignant … mon soutien … ma force … ma confiance : A toi … mon cher papa

A la lumière de ma vie … la femme qui m'a donner la vie : A toi mon adorable Mama

A l'âme de mes grands-parents jadi fodil et Yaya Zhira …que dieu les gardes dans son paradis.

A qui on a dit d'eux :

{سَنَشُدُّ عَضُدَكَ بِأَخِيكَ}

A ceux qui ont inlassablement tendu la main dans mes moments de faiblesse

Mes frères, Ziad et Abdellah que dieu les garde comme un côté constant pour moi

A celle qui a cru en mes capacités…qui se tient derrière moi comme mon ombre…ma sœur Esma.

A mes tantes et mon deuxième père, mon oncle

A ma famille en France … les plus fidèles

A mes cousines : Zineb, Sima, Lamis, Maroua, Lina, Tasnime, Meryouma, Soudjoud que j'aime la plus, et celle qui est toujours à mes côtés Amel

A mes cousins : Jawad, Yahia, Moha, Nasser, Ilyes Adem, et les sucres de la famille Naoufel et Abderrahmane.

A ma deuxième sœur Ihsene que j'adore pour tous ce qu'elle fait pour moi.

A celle qui a partagé avec moi les journées fatiguées et les nuits blanches ma binôme Feyrouz

A mes amis…Abir, Nesrine, Imane, Leila, et toutes mes amis

A mon soutien dans la vie…qui me rappelle ma force…mon ami Issam que dieu le protège pour moi

À toute ma famille et à toutes les personnes que je connais et que j'aime.

REMMACHE Hala ♥

Dédicace

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Louange à Dieu qui m'a donné la force, la volonté et le courage pour achever ce travail modeste. Je remercie Dieu, notre Créateur, qui m'a accordé cette grâce.

Je dédie ce travail à: Ma chère mère, source de tendresse et de lumière qui guide mes pas dans la vie, à toi, les plus sincères remerciements pour tes sacrifices, tes précieux conseils, ton soutien constant et ta présence continue dans ma vie.

Mon cher père, à qui je suis profondément reconnaissant pour ses efforts immenses, ses conseils et sa vigilance constante.

Mes chères sœurs : Kenza, Ibtissam, Rokia, et Ramissa.

Les enfants de mes sœurs : Fadi, Yanis, Tasnim, Nariman, Rasim, Abdelrahman et Amina.

À l'âme de mes grands-parents : mon grand-père Ammar, Zagdoud, et ma grand-mère Zaghda... Que Dieu leur accorde Sa miséricorde et les fasse entrer au Paradis. Et à ma chère grand-mère Louisa, que Dieu prolonge sa vie et la garde pour nous.

À celle qui a partagé avec moi les jours difficiles et les nuits blanche, ma binôme Hala. Mes chers amis : Aya, Amani, Chaima, Ikram, et toutes mes chères amies.

À mon soutien dans la vie, celui qui me rappelle toujours ma force, mon fiancé et mon pilier, Djahid, que Dieu le protège pour moi et pour sa famille.

À mon professeur et mentor, BOUTINE Rachid, qui a contribué à la réussite de ce travail.
À mes oncles et tantes, ainsi qu'à mes cousins et cousines.

Enfin, à toute ma famille et à toutes les personnes que je connais et que j'aime, je vous remercie du fond du cœur pour tout le soutien et l'aide que vous m'avez apportés pour la réalisation de ce travail.

Que Dieu vous bénisse tous et vous accorde tout le bien.

والسلام عليكم ورحمة الله وبركاته.

SAAD HAMIDECHE Feyrouz ♥

Tableau de Matière

Chapitre 01: Etat de l'art

Chapitre 02: Le modèle de boids de Reynolds et la méthode de champ de potentiel

Chapitre 03: Conception

Chapitre 04: Implémentation et Résultats

Table des Figures

Introduction générale

Introduction générale

La robotique tell qu'elle est connue aujourd'hui est une science interdisciplinaire qui englobe de nombreux domaines tels que la mécanique, la mécatronique, l'électronique, l'informatique et l'IA. Comprenant de vastes champs de recherche : la vision, la planification, la commande de mouvement, le design, etc. Cependant, l'un des problèmes les plus importants reste la coopération, la planification et la coordination des mouvements au sein d'une architecture de commande dans un contexte multi-robots. L'étude des systèmes multi-robots est devenue une préoccupation majeure dans le milieu de la recherche en robotique, car quelles que soient les capacités d'un robot unique il reste spatialement limité.

La robotique collective vise plus spécifiquement à concevoir des systèmes capables de se déplacer de façon collective. De nombreuses applications, tel que le vol des drones, domaine militaire, domaine des voiture autonomes …etc.

Depuis maintenant une trentaine d'années, la robotique collective est en plein développement. Afin de contrôler les comportements de groupes de robots, elle s'est souvent inspirée des extraordinaires comportements collectifs des insectes sociaux et des animaux. La nature a en effet développé de nombreuses stratégies permettant à ces animaux de résoudre collectivement des problèmes grâce à une organisation décentralisée et à une coordination des individus par des mécanismes auto-organisés. C'est le cas, par exemple, des comportements de suivi de piste de phéromones chez les fourmis, de sélection d'une source de nourriture chez les abeilles, et les comportements de volées d'oiseaux, etc.

Notre projet est basé sur l'inspiration biologique et plus précisément sur les volées d'oiseaux, ce phénomène permet de résoudre le problème d'évitement collectif d'obstacle. Grace à la méthode de champ de potentiel.

Notre travail se présente comme suit :

Chapitre 01 : Est consacré pour donner quelques définitions de la robotique et introduire le domaine, et présenter les travaux les plus récents dans le domaine.

Chapitre 02 : Présente la méthode proposée pour la navigation d'un système multi-robot basé sur le modèle de boids proposé par Reynolds, fusionner avec la méthode de champ de potentiel proposé par Oussama khatib.

Chapitre 03 : Présente la conception de notre projet en utilisant des organigrammes.

Chapitre 04 : Sera dédié pour présenter des cas de tests et analyse de résultats.

Chapitre 01:

Etat de l’art

1. Introduction

Dans ce chapitre, nous allons commencer par quelques définitions. Ensuite nous aborderons la notion d'évitement d'obstacle ainsi que ses différentes méthodes, nous allons faire aussi une représentation des approches bio inspirées. Enfin, nous ferons l'état de l'art de quelques techniques utilisées pour le déplacement d'un groupe de robots.

2. Définition

2.1. Robotique

La robotique est l'ensemble des techniques permettant la conception et la réalisation de machines automatiques ou de robots. Par extension, la robotique fait aussi référence à l'ensemble des domaines scientifiques et industriels en rapport avec la conception et la réalisation de robots. [1]

2.2. Robot

Un robot est un système alimenté en énergie qui évolue dans un environnement statique ou dynamique, il est formé d'un microcontrôleur ainsi que d'un ou plusieurs capteurs et actionneurs. [2]

2.3. Multi agent

Système complexe composé d'un environnement dans lequel des agents interagissent afin de résoudre un problème. [3]

3. La navigation

La navigation d'un robot mobile consiste à trouver un mouvement qui amène le robot d'une configuration initiale à une configuration finale. Généralement, ceci exige deux facultés : la planification de trajectoires et la réaction pour éviter des obstacles. [4]

3.1. Approches de planification en environnement statique

La planification de chemin est la formulation la plus simple du problème de planification de mouvement puisque l'environnement dans lequel évolue le robot est statique. Ainsi, le problème se réduit à l'aspect géométrique de la solution et par conséquent toute courbe qui assure les contraintes de continuité et de la non-collision est acceptable comme solution et le défi revient, en général, à calculer le plus court chemin entre la configuration initiale et la configuration finale. Plusieurs méthodes ont été développées et en fonction de la

connaissance à priori que le robot dispose de son environnement, trois familles d'approches se présentent :

3.1. A. Les approches globales (délibératives ou déterministes)

Le principe est de déterminer une solution complète du problème si elle existe, avant que le robot débute son déplacement, en se basant sur une connaissance aussi complète que possible de l'environnement de travail.

Parmi les approches, nous citons : la méthode de décomposition cellulaire (Approchées et exacte), les méthodes par Roadmaps (dont : les graphes de visibilité, la méthode des cônes généralisées, les diagrammes de voronoi ou encore la méthode des silhouettes) et la méthode de résolution de type rétraction… etc. [5]

3.1. B. Les approches locales (réactives)

Dans plusieurs applications, l'environnement n'est pas connu à priori. Dans ces conditions, il n'est pas possible de déterminer un mouvement complet jusqu'au but avant que le robot ne commence à se déplacer. La planification correspond alors à gérer dynamiquement les informations relatives à l'environnement. En effet, le principe est de calculer uniquement le mouvement à appliquer au prochain pas temporel et c'est à partir des données capteurs recueillies par le système robotique à chaque instant. Par conséquent, la représentation de l'environnement est construite au fur et à mesure du déplacement du robot.

Parmi les approches, nous citons : méthode par fonction de champ de potentiel, méthode de diagramme de proximité, la fenêtre dynamique, la logique floue, les Rapidly Exploring Random Trees(RRT), méthode bande élastique, les méthodes heuristiques,… etc. [5]

3.1. C. Les approches bio-inspires

Les approches bios inspirées sont les plus anciennes et les plus populaires, créées pour deux objectifs principaux. Le premier consiste à la modélisation des systèmes naturels et leur simulation sur ordinateur. Le deuxième implique l'étude des phénomènes naturels pour développer des systèmes informatiques et des algorithmes aptes à résoudre des problèmes complexes. Ces systèmes ou approches sont aussi connus sous le nom de métaphores biologiques.

3.1. C. 1. L'intelligence en essaim (IE)

Plusieurs définitions de l'intelligence en essaim peuvent être trouvées dans la littérature.

« L'intelligence en essaim comporte toute tentative de concevoir des algorithmes ou des outils distribués de résolution de problèmes, inspirée par le comportement collectif des insectes sociaux et d'autres sociétés animales ».

On peut accorder à cette intelligence collective une autre définition comme : « C'est l'auto organisation d'un groupe qui fait émerger un comportement inattendu ».

L'intelligence en essaim regroupe principalement deux axes de recherche : le premier appuie sur les travaux basés sur les insectes sociaux (fourmis, abeilles, sauterelles, termites, moustiques et insectes migrateurs), et le deuxième porte sur les travaux basés,sur la capacité des sociétés humaines à traiter des connaissances (les bactéries, les oiseaux, et les animaux aquatiques/poissons). Bien que les deux approches soient tout à fait différentes dans leur enchaînement d'étapes et leurs sources d'inspiration, ils présentent certains points communs. En général, les deux approches s'appuient sur une population (colonie ou essaim) d'individus (insectes sociaux ou particules) capables d'interagir (directement ou indirectement) entre eux à travers un environnement. On peut citer parmi ces approches : l'algorithme d'optimisation par essaim de particule (PSO) et l'algorithme d'optimisation par les colonies de fourmis (ACO).

- **Domaines d'application**

Elles sont conçues pour résoudre les problèmes d'optimisation discrète, d'ordonnancement séquentiel, d'affectation, du voyageur de commerce, du routage réseau, de déterminer la stratégie de contrôle, etc. [6]

3.1. C.2. Les systèmes immunitaires artificiels (SIA)

- **Principe**

Le système immunitaire (SI) est le responsable de nous protégé contre l'attaque des micro-organismes externes avec plusieurs mécanismes de défense, plusieurs niveaux, et une certaine redondance. Il inclut l'apprentissage et la mémoire.

Les systèmes immunitaires artificiels (SIA) incluent tout système ou outil de calcul qui extrait les idées et les métaphores du système biologique immunitaire. Les éléments de base d'un SIA sont :

- La maturation d'affinité : promouvoir l'apprentissage (l'adaptation) par le biais somatique d'hyper mutation et de sélection.
- La sélection clonale : décrit comment les cellules immunitaires et des molécules interagir avec les antigènes.
- La sélection négative : génère un ensemble(s) des détecteurs du non-soi pour la détection d'anomalie.
- Le Réseau immunitaire : effectue la dynamique et le méta dynamique du système structuré d'une manière en forme de réseau.
- La théorie du danger : des cellules présentatrices d'antigène (cellules dendritiques) du système immunitaire sont eux-mêmes activées via une alarme : les signaux de danger.

- **Domaines d'application**

Les SIA sont construits pour l'apprentissage machine et la RDF, la détection des anomalies et sécurité des systèmes informatiques, l'analyse des données (découvertes des connaissances des BDD, clustering..), les systèmes à base d'agent, la programmation la Planification, la navigation autonome, le contrôle, la recherche, l'optimisation…etc. Nous nous intéressons, dans cette thèse, à la technique de reconnaissance basée sur les systèmes immunitaires artificiels (SIA). Cette technique diffère des techniques déjà citées précédemment par le fait que les SIA ont la capacité de mémorisation et d'adaptation, et qui sont très utiles dans le domaine de reconnaissance d'écriture manuscrite. Le chapitre qui suit présente une description plus détaillée de ces systèmes. [6]

3.2. Evitement d'obstacle

Le suivi de la trajectoire planifiée ne permet pas de garantir l'absence de collision. En effet, des collisions peuvent se produire lors de l'exécution de la trajectoire, dues à :

- une localisation imparfaite,
- un plan imprécis,
- des obstacles qui n'étaient pas dans le modèle de l'environnement utilisé pour la planification de trajectoire.

Tous ces éléments font que le mouvement initialement planifié doit être adapté lors de son exécution, et que des stratégies d'évitement réactif d'obstacles doivent être mises en

œuvre. On adoptera des stratégies différentes en fonction du type de système, de sa vitesse, et du champ d'application. [7]

3.3. Suivi de trajectoire

Le suivi de trajectoire consiste à calculer les commandes des actionneurs du système permettant de réaliser le mouvement planifié. Un robot étant considéré comme un système dynamique, on utilise des méthodes de commande par retour d'état pour asservir le système sur une trajectoire de référence. [7]

4. Quelques techniques utilisées pour la navigation d'un groupe de robots

Il existe différentes techniques utilisées pour la navigation d'un groupe de robots

4.1. La logique floue

4.1.1. Définition

- La logique floue, aussi appelée théorie des ensembles flous, est une branche des mathématiques et de l'informatique qui permet de traiter des concepts et des situations imprécises ou incertaines en utilisant des valeurs de vérités continues plutôt que binaires (vrai ou faux).
- La logique floue (ou fuzzy logic en anglais) est une approche de l'informatique basée sur des « degrés de vérité » plutôt que sur la logique booléenne habituelle « vrai ou faux » (1 ou 0) sur laquelle repose l'informatique moderne. Elle a été formulée pour la première fois par le Dr Lotfi Zadeh, de l'université de Californie à Berkeley. [8]

4.1.2. Différence entre la logique floue et la logique classique

La logique classique se base sur le principe de non-contradiction et qui attribue une valeur de vérité précise à chaque proposition (soit vrai, soit faux : 1 ou 0) alors que la logique floue permet d'exprimer des nuances et des degrés de vérité. Pour cela la logique floue se propose de remplacer les variables booléennes par des variables flous.

- ***Exemple de navigation d'un groupe de robots en utilisant la logique floue*** [Projet de Miha Moskon, Miha Mraz, Nikolaj Zimic, Iztok Lebar Bajec ; University of Ljubljana, Faculty of Computer and Information Science 1000 Ljubljana, Trza ˇ ska 25, Slovenia] [9] :

Les développeurs de ce projet présentent un modèle basé sur la logique floue pour la simulation du comportement de recherche de nourriture chez les oiseaux. Le cœur du modèle repose sur le modèle flou pour la simulation informatique du vol en groupe d'oiseaux présenté par Lebar Bajec et al. Ce dernier a été étendu de manière à permettre la simulation du comportement de recherche de nourriture des oiseaux. Afin de moderniser le modèle original, il a été nécessaire d'avancer dans le monde artificiel et l'oiseau synthétique (synbird). Le premier a été élargi avec l'introduction de zones d'alimentation des régions circulaires contenant de la nourriture. Tout comme les synbirds, elles ont été implémentées en tant qu'animats, c'est-à-dire au moyen d'une fonction de transition en trois étapes. La première étape de cette fonction est responsable de la sélection d'informations sur les synbirds actuellement dans la zone, la deuxième pour calculer le changement de la nourriture disponible, et la troisième pour calculer le nouvel état de la zone d'alimentation. En plus de l'introduction des zones d'alimentation, ils ont amélioré les synbirds avec la notion de faim.

Pour ce faire, deux impulsions ont été ajoutées aux impulsions de base du modèle et l'état interne du synbird a également été modifié. Pour soutenir l'attraction des zones d'alimentation, une nouvelle fonction de perception a été ajoutée et la sélection des actions a été améliorée afin de prendre en compte les influences des impulsions nouvellement introduites. Le comportement des synbirds est régi par leur type de comportement, qui peut être l'alimentation, le décollage, le vol ou l'atterrissage. Alors que l'impulsion du type de comportement est responsable de la transition entre les types de comportement, l'impulsion d'alimentation, par rapport au type de comportement actuel, influence la faim, l'altitude de vol, la vitesse et la direction du vol. Avec les extensions apportées, ils ont rapproché le modèle d'une simulation plus naturaliste du comportement des oiseaux.

Figure 1- 1 *:* Bande d'oiseaux [25]

4.2. Flux d'essaim

- ***Exemple de navigation d'un groupe de robots en utilisant le Flux d'essaim*** **[Projet de Alexandre Bonnefond, Olivier Simonin ; Univ. Lyon, Inria, INSA de Lyon, CITI & LIP Labs, 6 Av. des Arts 69621 Villeurbanne cedex] [10] :**

Les chercheurs proposent des extensions aux modèles de flocking existants afin d'améliorer leurs performances dans des environnements ayant des obstacles impactant les communications ainsi que les trajectoires des agents. En effet, les contraintes imposées par les obstacles sont généralement la cause de coupures de communication menant souvent à la séparation de la flotte en plusieurs clusters. Dans ce contexte, ils étendent deux modèles standards afin de renforcer leurs capacités à rester connectés dans des environnements avec déférentes distributions d'obstacles. En tenant compte de la propagation radio, ils modélisent comment les obstacles impactent les communications dans un simulateur qu'ils utilisent notamment pour optimiser les paramètres du flocking. Les résultats des simulations montrent l'efficacité des modèles proposés et la façon dont ils s'adaptent à ces nouvelles contraintes environnementales.

4.3. Intelligence artificielle

4.3.1. Définition

L'intelligence artificielle (IA) est une branche de l'informatique dédiée à la conception de machines capables d'imiter le cerveau humain dans des tâches telles que l'apprentissage ou le raisonnement. Découvrez les bases de son fonctionnement et les bénéfices que peuvent en tirer les entreprises, notamment en matière d'exploitation et de business intelligence. [11]

- ***Exemple de navigation d'un groupe de robots en utilisant l'intelligence artificielle*** **[Projet de Fredy Mart´ınez, Holman Montiel, Luis Wanumen ; Facultad Tecnologica, Universidad Distrital Francisco Jos ´ e de Caldas, Bogot ´ a D.C, Colombia] [12] :**

Les comportements sociaux chez les animaux tels que les abeilles, les fourmis et les oiseaux ont montré des niveaux élevés d'intelligence d'un point de vue du système multi-agent. Ils présentent des solutions viables à des problèmes du monde réel, en particulier dans la navigation dans des environnements contraignants avec des plates-formes robotiques simples. Parmi ces comportements, ils trouvent le regroupement d'essaims, qui a été

largement étudié à cette fin. Des algorithmes de regroupement ont été développés à partir de règles comportementales de base, qui nécessitent souvent un réglage de paramètres pour des applications spécifiques. Cependant, l'absence d'une formulation générale pour le réglage a rendu difficile la mise en œuvre de ces stratégies dans diverses conditions réelles, et même pour reproduire les comportements en laboratoire. Dans cet article, ils proposent un schéma de regroupement pour de petits robots autonomes capables d'apprendre de manière autonome dans des environnements dynamiques, dérivé d'un processus d'apprentissage en profondeur par renforcement. Leur approche permet le regroupement indépendamment de la taille de la population et des caractéristiques de l'environnement, avec une intervention externe minimale. Leur modèle de système multi-agent considère l'action de chaque agent comme une fonction linéaire ajustant dynamiquement le mouvement en fonction des interactions avec d'autres agents et de l'environnement. Leur stratégie est une contribution importante à la mise en œuvre du regroupement dans le monde réel. Ils démontrent que leur approche permet le regroupement autonome dans le système sans nécessiter de réglage spécifique des paramètres, ce qui la rend idéale pour des applications où il est nécessaire que des plates-formes robotiques simples naviguent dans des environnements dynamiques. (Voir la figure 1-2).

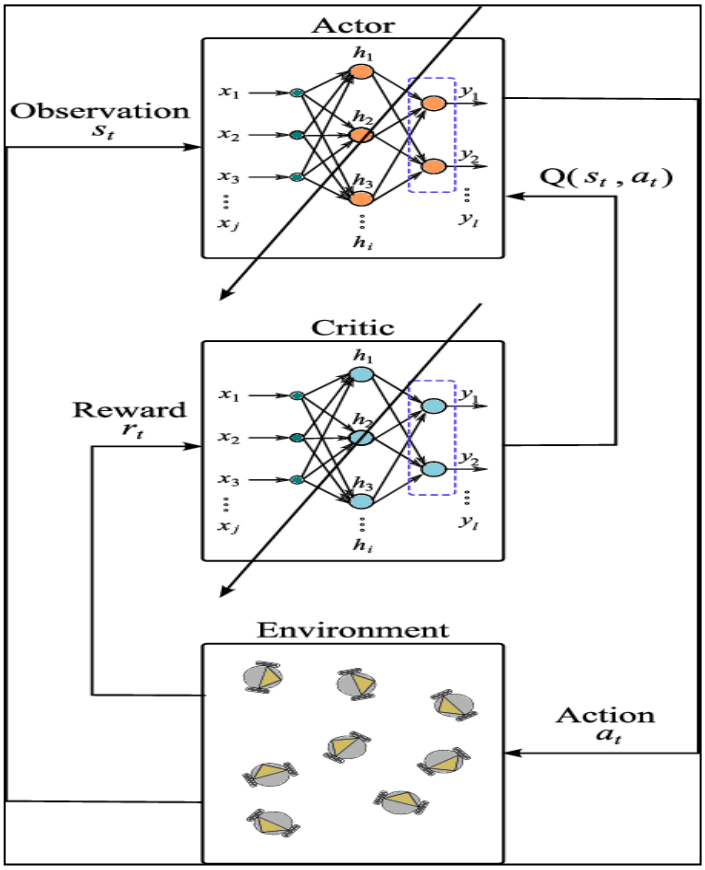


Figure 1- 2: Architecture du dispositif d'apprentissage proposé [12]

4.4. Système multi-agent

- ***Exemple de navigation d'un groupe de robots en utilisant système multi-agent*** **[Projet de Andreas Brandstatter, Radu Grosu, CPS, Technische Universitat Wien (TU Wien), Austria, Department of Computer Science, Stony Brook University, USA] [13] :**

Ils introduisent le concept novateur du Contrôle Prédictif Spatial (SPC) pour résoudre le problème suivant : étant donné une collection d'agents (par exemple, des drones) avec des contrôleurs bas niveau de position (LLC) et une fonction de coût distribuée spécifique à la mission, comment un contrôleur distribué peut-il atteindre et maintenir la minimisation de la fonction de coût sans un modèle de plante et uniquement avec des observations de position de l'environnement ? Leur contrôleur SPC entièrement distribué est basé strictement sur la position de l'agent lui-même et sur celles de ses agents voisins. Ces informations sont utilisées à chaque pas de temps pour calculer le gradient de la fonction de coût et effectuer une anticipation spatiale pour prédire la meilleure position cible suivante

pour le LLC. En utilisant un environnement de simulation haute-fidélité, ils montrent que le SPC surpasse la classe de contrôleurs la plus étroitement liée, les Contrôleurs de Champ Potentiel, sur le problème de la formation d'essaim de drones. Ils démontrent également que le SPC est capable de faire face à un éventuel écart de transfert de la simulation au monde réel en présentant ses performances sur du matériel réel, à savoir leur mise en œuvre de la formation d'essaim avec neuf drones Crazyflie 2.1.

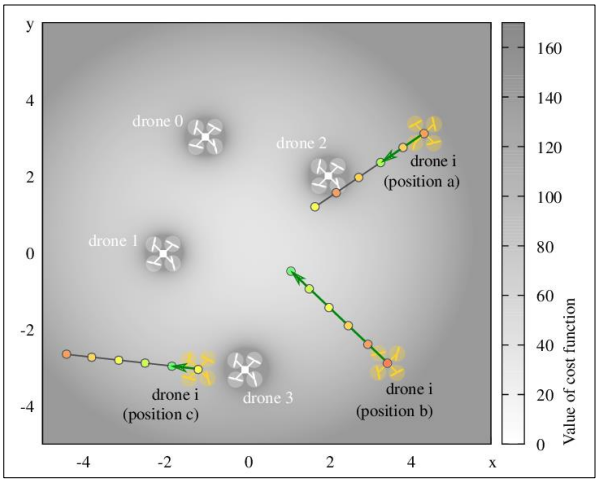


Figure 1- 3: Pour un ensemble donné de drones voisins (drones 0 à 3 en blanc), le la fonction de coût pour le drone i est affichée sous forme de carte thermique en niveaux de gris. La direction dans lequel le drone i (en jaune) doit se déplacer est déterminé par la pente de la fonction de coût (pour les positions a, b et c). La SPC évalue le coût au l'anticipation spatiale (indiquée par des points colorés) dans cette direction et choisit la meilleure valeur pour la prochaine position du drone i (flèche verte). [13]

5. Conclusion:

La navigation d'un groupe de robot est un défi complexe qui nécessite l'utilisation de méthodes appropriées. Nous avons découvert diverses méthodes telles que La logique floue, Flux d'essaim, Intelligence artificielle et Système multi-agent. Le choix de la technique dépendra des spécificités du problème et des objectifs à atteindre, et il est nécessaire de prendre en compte la complexité de l'environnement lors du choix d'une méthode d'évitement.

Chapitre 02 :
Le modèle de boids de Reynolds et la méthode de champ de potentiel

1. Introduction

Dans ce chapitre, nous parlerons d'un des modèles les plus fascinants dans le domaine de l'intelligence artificielle et de l'informatique, le modèle de boids de Reynolds. Nous discuterons de l'origine de l'idée et de la manière dont elle a été développée, en plus d'une explication simplifiée des règles de Reynolds qui constituent la base de ce modèle. En fin nous parlerons du principe d'évitement collectif d'obstacle par champs de potentiel.

2. Le modèle de boids de Reynolds

Nous allons faire un petit bon dans le temps pour revenir en **1986. Craig Reynolds** travaille alors comme graphiste chez Symboliques, une entreprise qui fabrique des ordinateurs. Tous les jours, lors de sa pause déjeunée, il a la chance d'admirer des nuées d'oiseaux volé dans le ciel.

Il faut savoir qu'à cette époque, l'informatique fait de nombreux progrès dans le domaine de la vie artificielle**.** C'est notamment à cette époque que John Conway invente le jeu de la vie, et que Christopher Langton développe sa fourmi informatique. Craig Reynolds va donc se demander s'il est possible de faire la même chose pour simuler le mouvement collectif de ces oiseaux.

Cette question était alors un sérieux problème sur lequel se heurtaient les biologistes, et là où les scientifiques de l'époque faisaient l'erreur de chercher à identifier un chef de groupe, Reynolds va, au contraire, imaginer un fonctionnement plutôt décentralisé, dans lequel un comportement collectif émerge à partir des interactions entre les oiseaux. Après quelques recherches, l'informaticien découvre que trois règles comportementales très simples suffisent. Ils baptiseront alors ses oiseaux numériques les « **Boids** ». [14]

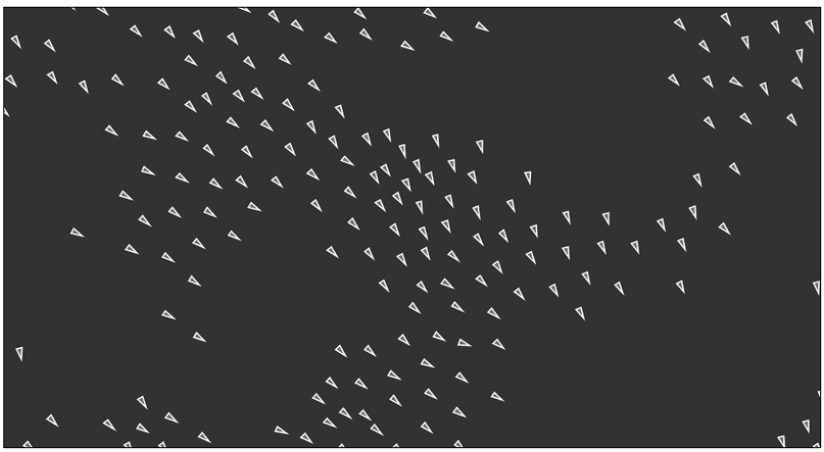


Figure 2- 1: Les boids de Reynolds [26]

Un modèle comportemental légèrement plus élaboré a été utilisé dans les premières expériences. Cela comprenait l'évitement prédictif des obstacles et la recherche d'objectifs. L'évitement d'obstacles a permis aux boids de voler à travers des environnements simulés tout en évitant les objets statiques. Pour les applications en animation par ordinateur, un comportement de recherche d'objectif de faible priorité a amené le troupeau à suivre un chemin scripté. [15]

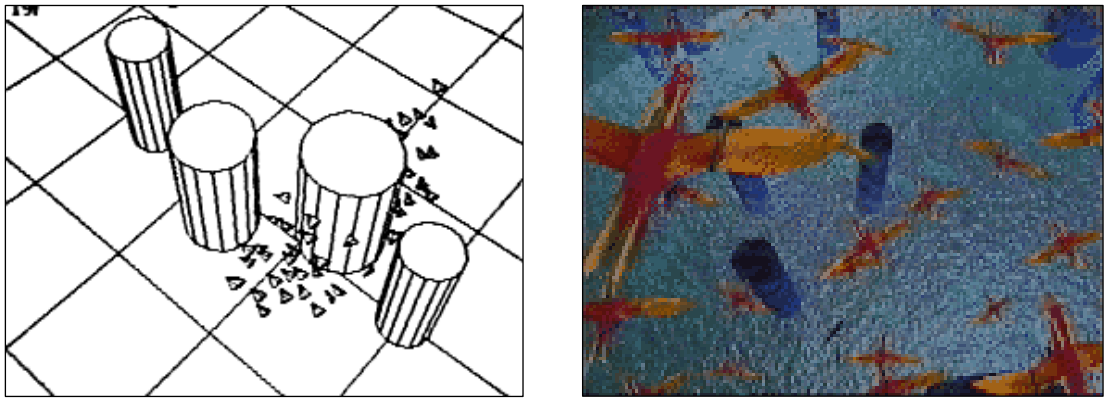


Figure 2- 2: Un ensemble de boid simulé évitant des obstacles cylindriques(1986). [15]

3. Les règles de comportement :

Dans son expérience, Reynolds a utilisé 3 règles de conduites (cohésion, séparation et alignement), qui sont :

3.1. La cohésion

Est un comportement qui amène les agents à se diriger vers le « centre de masse », c'est-à-dire la position moyenne des agents dans un certain rayon. Le centre de masse est simplement la position moyenne de tous les boids Cependant, le centre de masse est une propriété de tout la bande entière ; ce n'est pas quelque chose qui serait examiné par un boid individuel. Il est préfère de déplacer le boid vers son centre perçu, qui est le centre de tous les autres boids, sans compter lui-même. Après avoir calculé le centre perçu, nous devons déterminer comment déplacer le boid vers lui. Pour le déplacer de 1% vers le centre ceci est donné par **(pc_J – b_J.position) / 100**. (Voir la figure 2-3). [16]

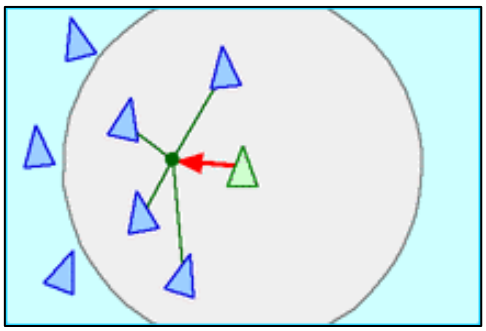


Figure 2- 3: La cohésion [16]

3.2. La séparation

Est le comportement qui amène un agent à s'éloigner de tous ses voisins. Le but de cette règle est de permettre aux boids de s'assurer qu'ils n'entrent pas en collision les uns avec les autres. Et s'il se trouve à une petite distance définie (disons 100 unités) d'un autre boid, il lui encore loin qu'il est déjà.

Ceci est fait en soustrayant d'un vecteur **c** le déplacement de chaque boid proche. Nous initialisons **c** à zéro car nous voulons que cette règle nous donne un vecteur qui, lorsqu'il est ajouté à la position actuelle, éloigne d'un boid ceux qui lui sont proches. [16]

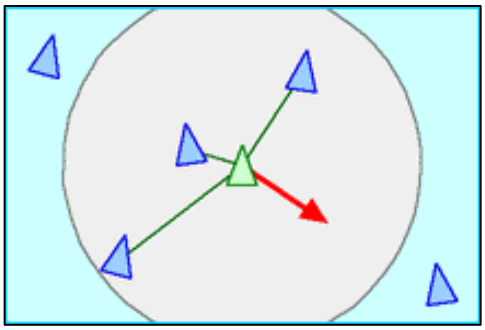


Figure 2- 4: La séparation [16]

3.3. L'alignement

Est un comportement qui amène un agent particulier à s'aligner avec des agents à proximité. Ceci est similaire à la règle 1, cependant, au lieu de faire la moyenne des positions des autres boids, nous faisons la moyenne des vitesses. Nous calculons une vitesse perçue, $\mathbf{pv_j}$, puis ajoutons une petite partie (environ un huitième) à la vitesse actuelle du boid. (Voir la figure 2-5). [16]

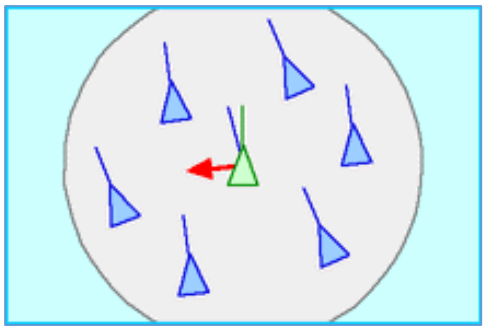


Figure 2- 5: L'alignement [16]

Chaque boid a un accès limite à la description géométrique de l'ensemble de la scène, mais le flocage nécessite qu'il réagisse uniquement aux camarades du troupeau dans un certain petit quartier autour de lui. Le voisinage est caractérisé par une distance (mesurée à partir du centre du boid) et un angle, mesuré à partir de la direction de vol du boid. Les compagnons de troupeau en dehors de ce quartier local sont ignorés. Le quartier pourrait être considéré comme un modèle de perception limitée (comme par les poissons dans les eaux troubles), mais il est probablement plus correct de le considérer comme définissant la région dans laquelle les troupeaux influencent la direction d'un boids.

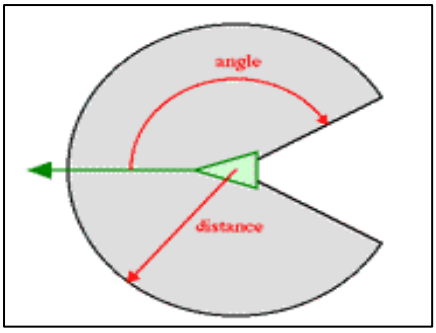


Figure 2- 6: Le voisinage local d'un boid [16]

4. La méthode de champ de potentiel

La méthode des champs de potentiel est bien connue comme outil pour la navigation des robots mobiles. Cette méthode est très intuitive, elle est efficace dans le sens où elle permet de contrôler la vitesse du robot pour qu'il atteigne sa position finale, et ce, relativement facile, en additionnant les différents effets intervenant sur le robot. [17]

4.1. Historique

La méthode du potentiel a été initialement introduite par Oussama. Khatib pour des robots manipulateurs. Elle diffère sensiblement des autres méthodes, car elle ne résulte pas d'un raisonnement purement géométrique. Par ailleurs, elle est peu satisfaisante en termes de planification, parce qu'elle n'est pas complète, ce qui signifie qu'il peut exister une solution sans que l'algorithme ne la trouve .Ceci étant, elle fournit une technique de navigation simple et compatible avec les exigences du temps réel. [17]

4.2. Principe de la méthode

Elle permet de considérer les robots comme des points sous l'influence d'un champ de potentiel artificiel. Celui-ci est créé par l'ensemble des objets présents sur le terrain (les murs, le robot, les obstacles et le point but **(B)).** Ce concept engendre un mouvement pour le robot qui est semblable à celui d'une balle qui dévalerait une colline en évitant les divers obstacles qui se présenteraient sur sa route (**O1** & **O2**). Ainsi, l'objectif (point but **(B))** du robot sera modélisé par une force attractive **(FA),** alors que les obstacles sur le terrain seront caractérisés par une force répulsive (**FR1** & **FR2**) ; De même, les bords du terrain joueront le rôle d'obstacles (force répulsives) vis-à vis du robot. [17]

Afin de déterminer le champ potentiel en un point, il faut établir le rôle de chaque objet sur le terrain, sommer ponctuellement l'effet de chacun, et finalement transformer cet effet en une force (**FB**) qui sera celle qui agira sur le robot et le fera bouger selon une direction bien précise.

Vous retrouvez dans la figure un schéma explicatif du principe :

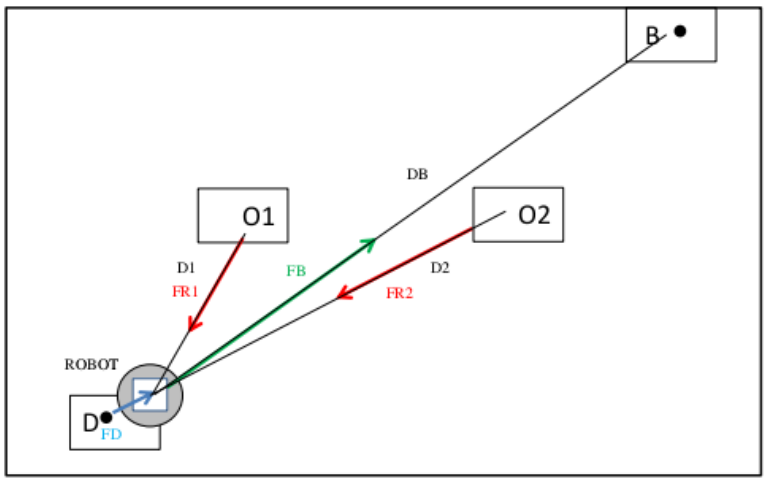


Figure 2- 7: Principe de la méthode champ de potentiel : auteur2024

FB=DB : force d'attraction du but

FR1=1/D1 : force de répulsion de l'obstacle 1

FR2=1/D2 : force de répulsion de l'obstacle 2

FD= 1/DD : force de répulsion de point de départ

Par exemple, supposons qu'il n'y ait aucun obstacle dans l'environnement et que le robot doive poursuivre cet objectif. Dans la planification conventionnelle, il faut calculer la position relative du robot par rapport à l'objectif, puis appliquer les forces appropriées qui conduiront le robot vers l'objectif.

Dans l'approche du champ potentiel, nous créons un champ attractif entrant à l'intérieur du but. Le champ potentiel est définis sur tout l'espace libre, et à chaque pas de temps, nous calculons le champ potentiel à la position du robot puis calculons la force induite par ce champ. Le robot doit alors se déplacer selon cette force.

La figure suivante illustre ce concept :

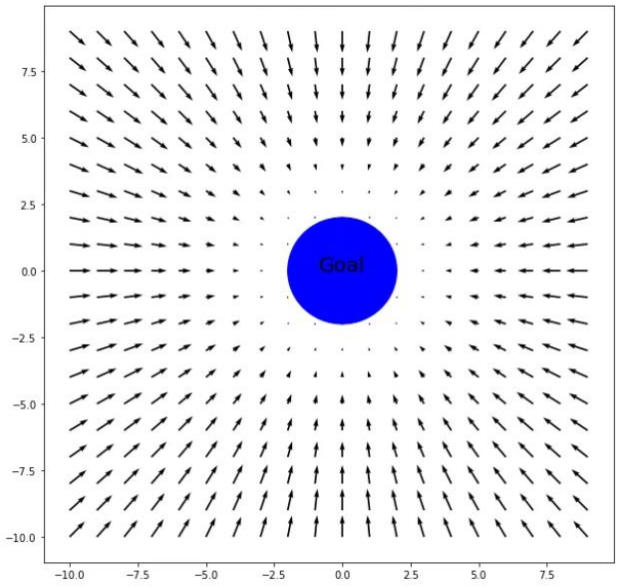


Figure 2- 8: Champ potentiel du but [15]

On peut également définir un autre comportement qui permet au robot d'éviter les obstacles. Nous faisons en sorte que chaque obstacle génère un champ répulsif autour de lui. Si le robot s'approche de l'obstacle, une force répulsive agira sur lui, l'éloignant de l'obstacle.

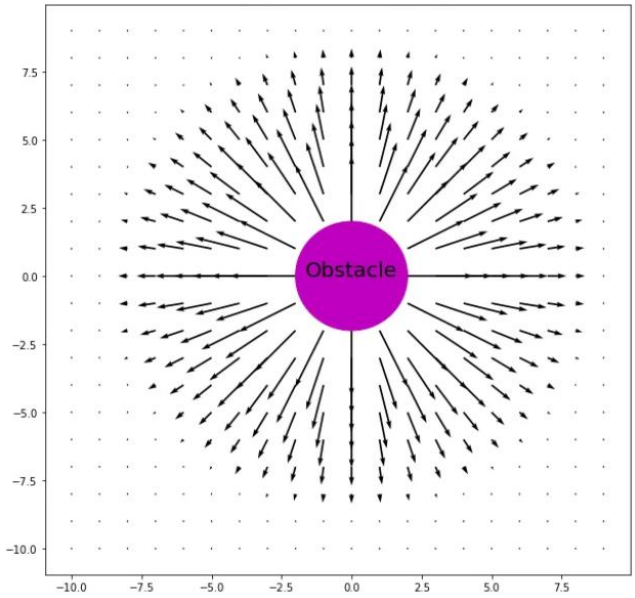


Figure 2- 9: Champ potentiel d'obstacle [15]

Les deux comportements, rechercher et éviter, peuvent être combinés en combinant les deux champs potentiels ; le robot peut alors suivre la force induite par le nouveau champ pour atteindre le but tout en évitant l'obstacle :

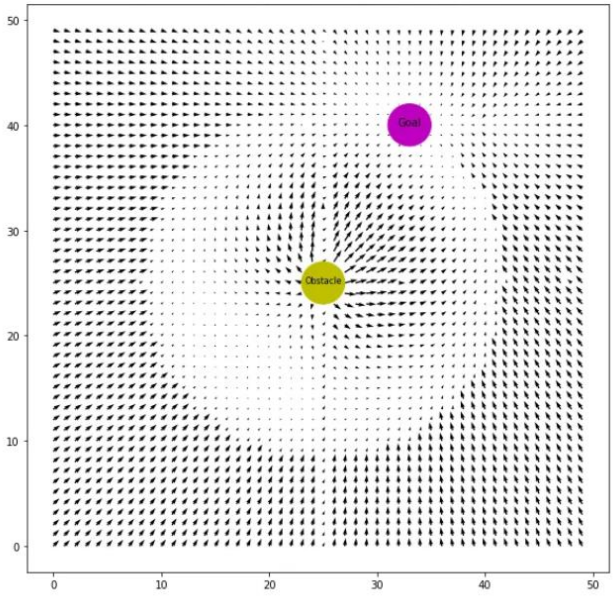


Figure 2- 10: Potentiel combiné lorsque le but et l'obstacle sont différents [15]

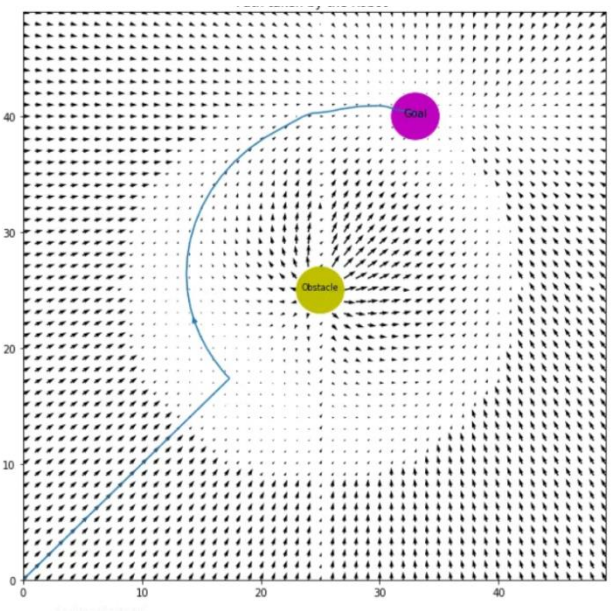


Figure 2- 11: Chemin emprunté par le robot [15]

Voyons maintenant les intrigues

S'il n'y a pas d'obstacle :

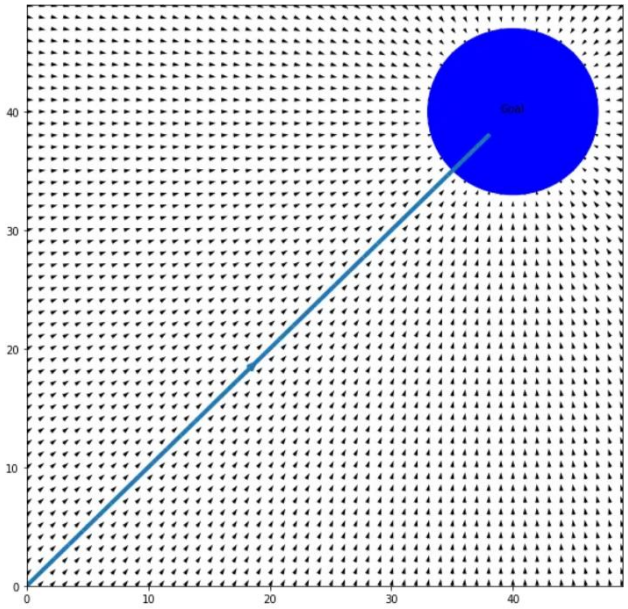


Figure 2- 12: Chemin du robot avec aucun obstacle [15]

S'il y a au moins un obstacle

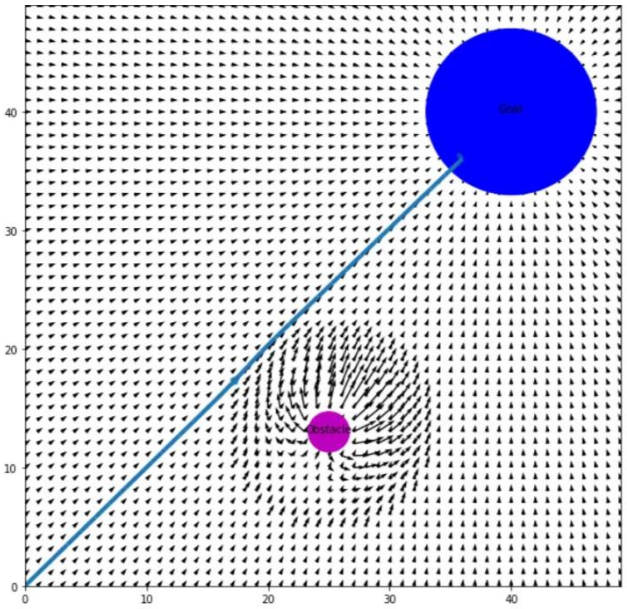


Figure 2- 13: Chemin du robot avec un obstacle [15]

S'il y a deux obstacles :

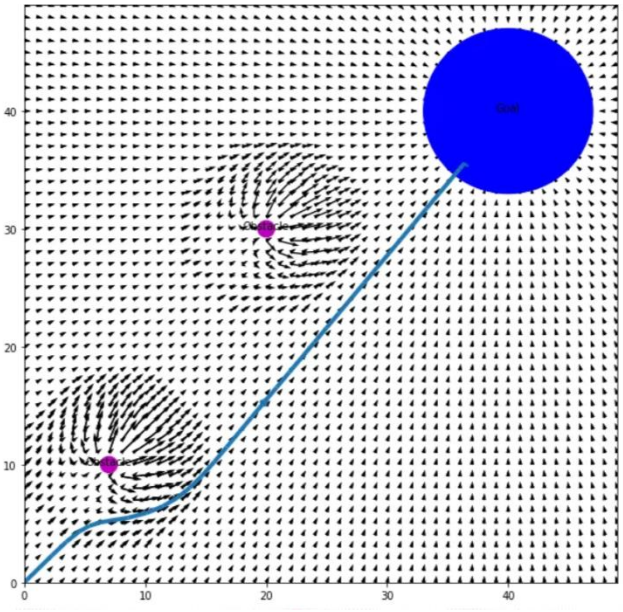


Figure 2-14: Chemin du robot avec 2 obstacles [15]

S'il y a plusieurs obstacles :

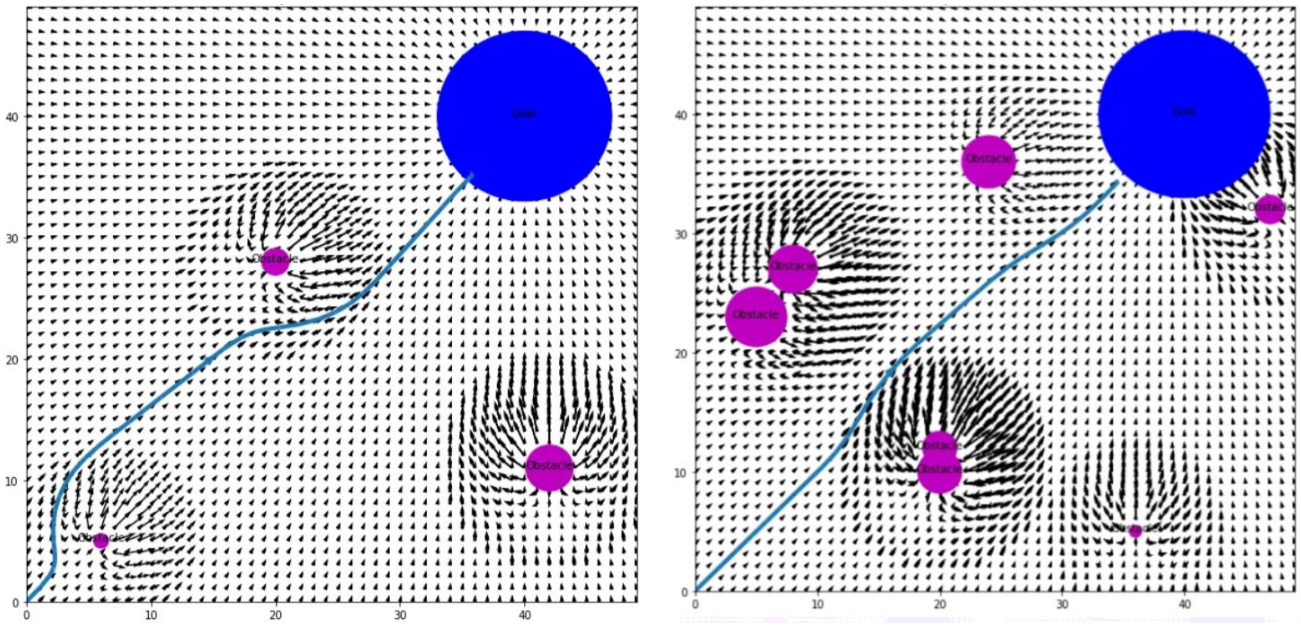


Figure 2-15: Chemin du robot avec plusieurs obstacles [15]

4.3. La Force attractive

La fonction de champ potentiel attractif est définie comme :

$$U_{att}(\phi) = \frac{1}{2}k_a(\phi - \phi_g)^2, \quad \textbf{(1)}$$

Où $\boldsymbol{\phi} = (\boldsymbol{x}, \boldsymbol{y})^{\boldsymbol{T}}$est la coordonnée du robot, $\mathbf{k_a}$ est le coefficient constant du champ attractif, et $\boldsymbol{\varphi_g}$ est la coordonnée du point cible. Le gradient négatif du champ de potentiel attractif,

La fonction est définie comme la fonction de force attractive, qui est :

$$\mathbf{F_{att}(\varphi) = -\nabla\,(U_{att}(\varphi)) = k_a\,(\varphi_g - \varphi),} \quad \textbf{(2)}$$

Où $\mathbf{F_{att}(\varphi)}$ est un vecteur dirigé vers $\boldsymbol{\varphi_g}$, qui est une corrélation linéaire avec la distance de $\boldsymbol{\varphi}$ à $\boldsymbol{\varphi_g}$. [19]

4.4. La Force répulsive

Les obstacles fournissent une force répulsive et repoussent le robot. Lorsque le robot est suffisamment éloigné des obstacles, les obstacles n'affecteront pas le mouvement du robot. Comme le montre l'équation (3), le champ de potentiel de répulsion la fonction peut être définie comme :

$$U_{req}(\phi) = \begin{cases} \frac{1}{2}k_r(\frac{1}{d(\phi)} - \frac{1}{d_{max}})^2, & d(\phi) \le d_{max}, \\ 0, & d(\phi) > d_{max}, \end{cases} \quad \textbf{(3)}$$

Où $\mathbf{k_r}$ est le coefficient constant du champ de répulsion, $\mathbf{d_{max}}$ est l'étendue d'impact maximale de l'Obstacle unique, et $\mathbf{d(\varphi)}$ est la distance actuelle entre le robot et l'obstacle. Le répulsif la force est le gradient négatif de la fonction de potentiel répulsif, qui est

$$\mathbf{F_{rep}(\varphi) = -\nabla\,(U_{rep}(\varphi)),} \quad \textbf{(4)}$$

Le champ potentiel répulsif total est la somme des champs potentiels répulsifs de tous les obstacles. [19]

4.5. La Force total

La fonction de champ potentiel finale est la suivante : [19]

$$\mathbf{U(\varphi) = U_{att}(\varphi) + \sum U_{rep}(\varphi)} \quad \textbf{(5)}$$

La force résultante sur le robot est :

$$\mathbf{F(x) = F_{att}(x) + \sum F_{rep}(x)} \quad \textbf{(6)}$$

4.6. Inconvénient de la méthode

Le champ de forces est un outil puissant mais sa grande faiblesse est qu'il entraine des problèmes de minima locaux comme c'est le cas dans la figure ci-dessous :

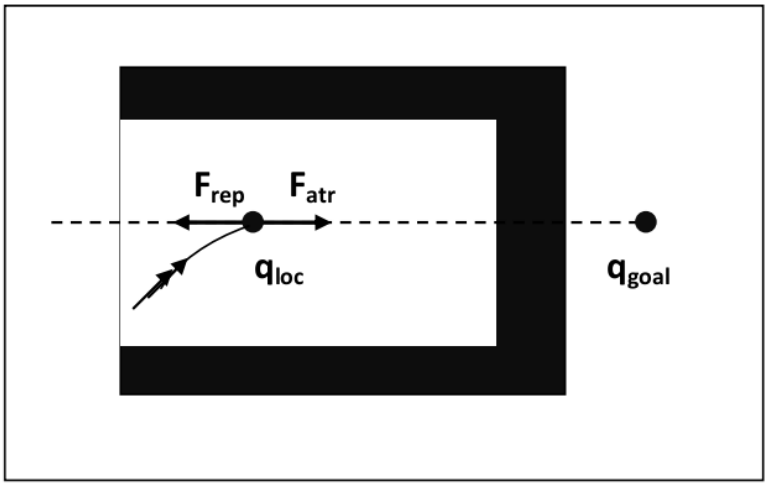


Figure 2-16: Exemple de minimum local [17]

Ce problème est une cause sérieuse de l'inefficacité de cette technique, où le robot peut coller dans un minimum local de la fonction potentielle différente de la configuration de but.

Dans cette configuration (endroit de q le robot est fermé à clé tout le temps en même position, parce que la force d'attraction et de répulsion sont équivalentes mais avec la direction opposée, c'est pourquoi toute la force résultante est égale à zéro et le robot ne peut pas changer en une autre position). [17]

5. Conclusion

Les grands laboratoires en robotique ont utilisé la méthode de planification de trajectoire par champs de potentiel pour les robots car elle présente des avantages par rapport à d'autres méthodes.

La méthode de champs de potentiels pour la planification des robots a été choisie pour la planification du robot franco-japonaise HRP-2 le robot QRIO le robot Nao et le robot iCub dans un environnement connu. [17]

Cependant, le problème de la navigation collective des robots dans un environnement inconnu sans collision, ce qui nécessite des méthodes efficaces. Pour atteindre cet objectif, nous avons choisi le modèle de boid de Reynolds.

Nous avons également choisi la méthode du champ potentiel. Où chaque obstacle est représenté comme une source de champ de potentiel. Les obstacles créent des zones où le champ de potentiel est élevé, ce qui génère des forces répulsives sur le robot lorsqu'il s'approche de ces zones.

Le champ de force potentiel autour de la destination attire le robot vers celle-ci, En effet lorsque les champs potentiels propagés par l'obstacle (répulsifs) et la destination (attractifs) sont fusionnés, le robot peut calculer un chemin sûr pour atteindre sa destination tout en évitant les obstacles sur son chemin.

Chapitre 03 :

Conception

1. Introduction

Dans ce chapitre, nous avons conçu le système multi-robots proposé en se basant sur les principes étudiés dans les chapitres précédents, en établissant des schémas décrivant les différentes étapes du système.

2. Schéma de fonctionnement d'un boid

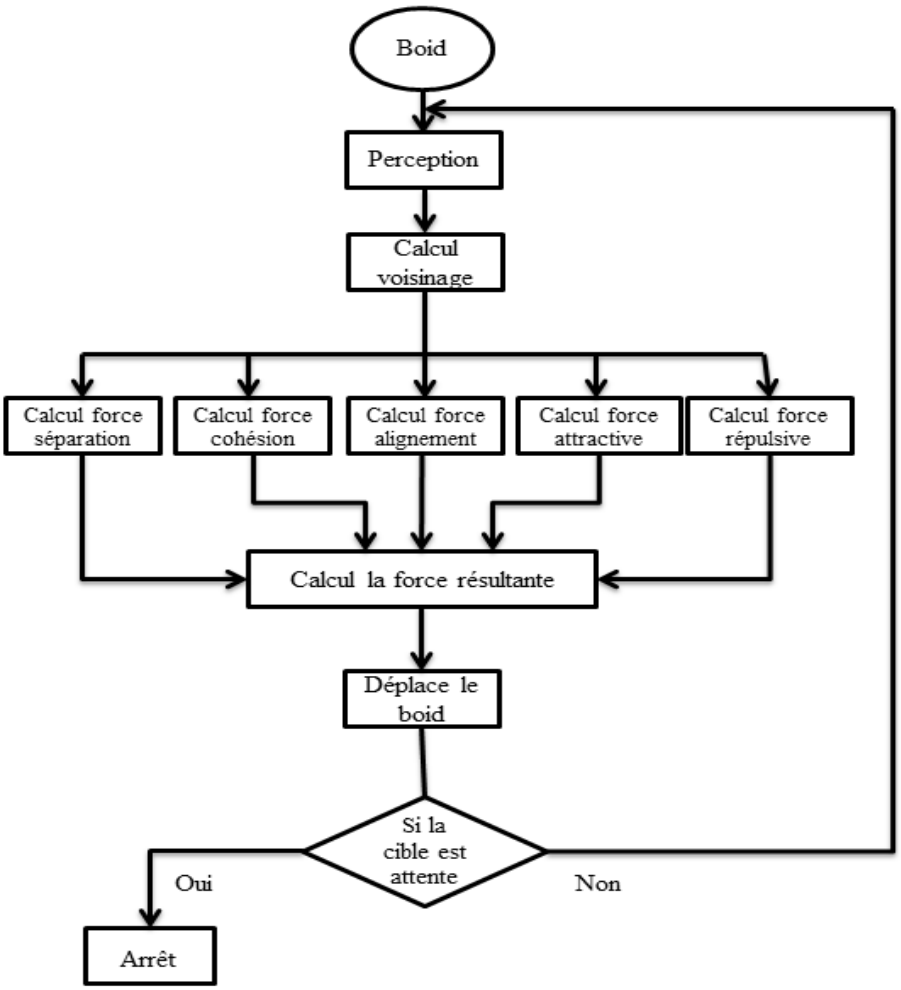


Figure 3- 1: Organigramme fonctionnement de boid : auteur 2024

3. Cohésion

3.1. Le pseudo code

Procédure règle 1(boid b_J)

Vecteur pc_J

Pour chaque boid b
Si b != b_J alors
$pc_J = pc_J + b.position$
Fin si
Fin pour

$pc_J = pc_J / N-1$

Return $(pc_J – b_J.position) / 100$

Fin procedure

3.2. Organigramme de cohésion

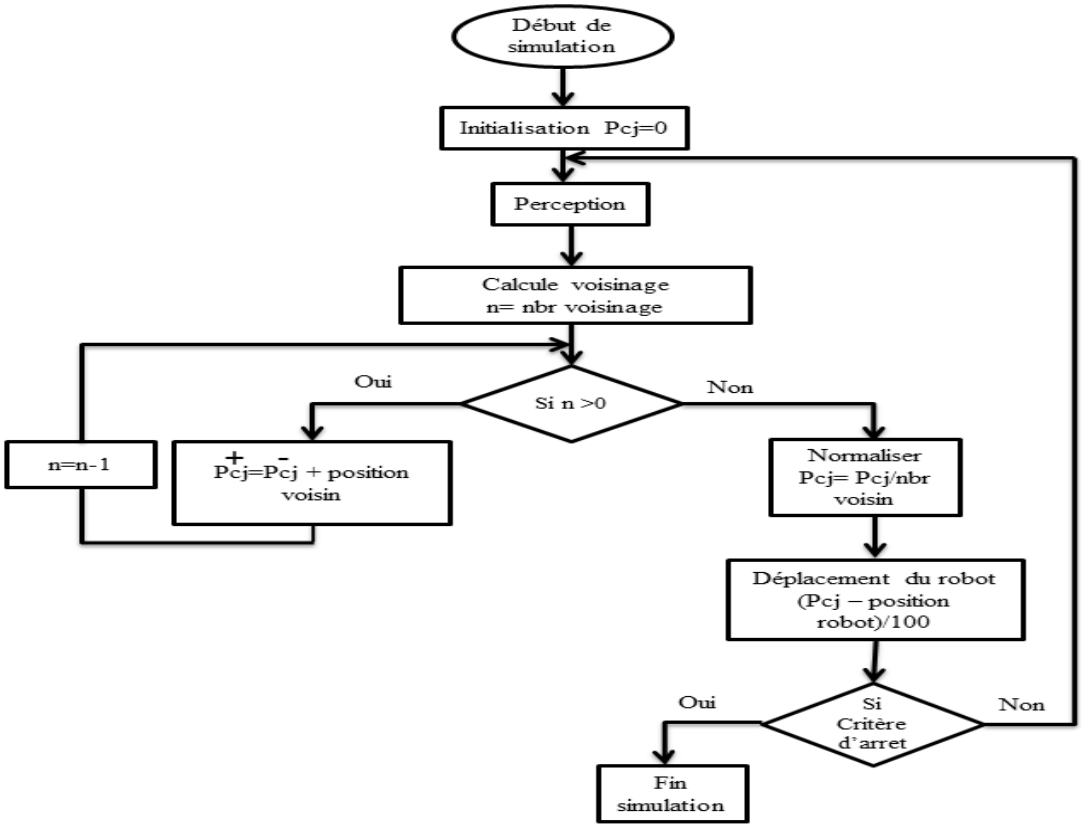


Figure 3- 2: Organigramme de cohésion : auteur 2024

4. Séparation

4.1. Le pseudo code

Procédure règle 2(boid b_J)

Vecteur c = 0 ;

Pour chaque boid b
Si b != b_J alors
Si |b.position – b_J.position| < 100 alors
c = c – (b.position – b_J.position)
Fin si
Fin si
Fin pour

Return c

Fin Procédure

4.2. Organigramme de séparation

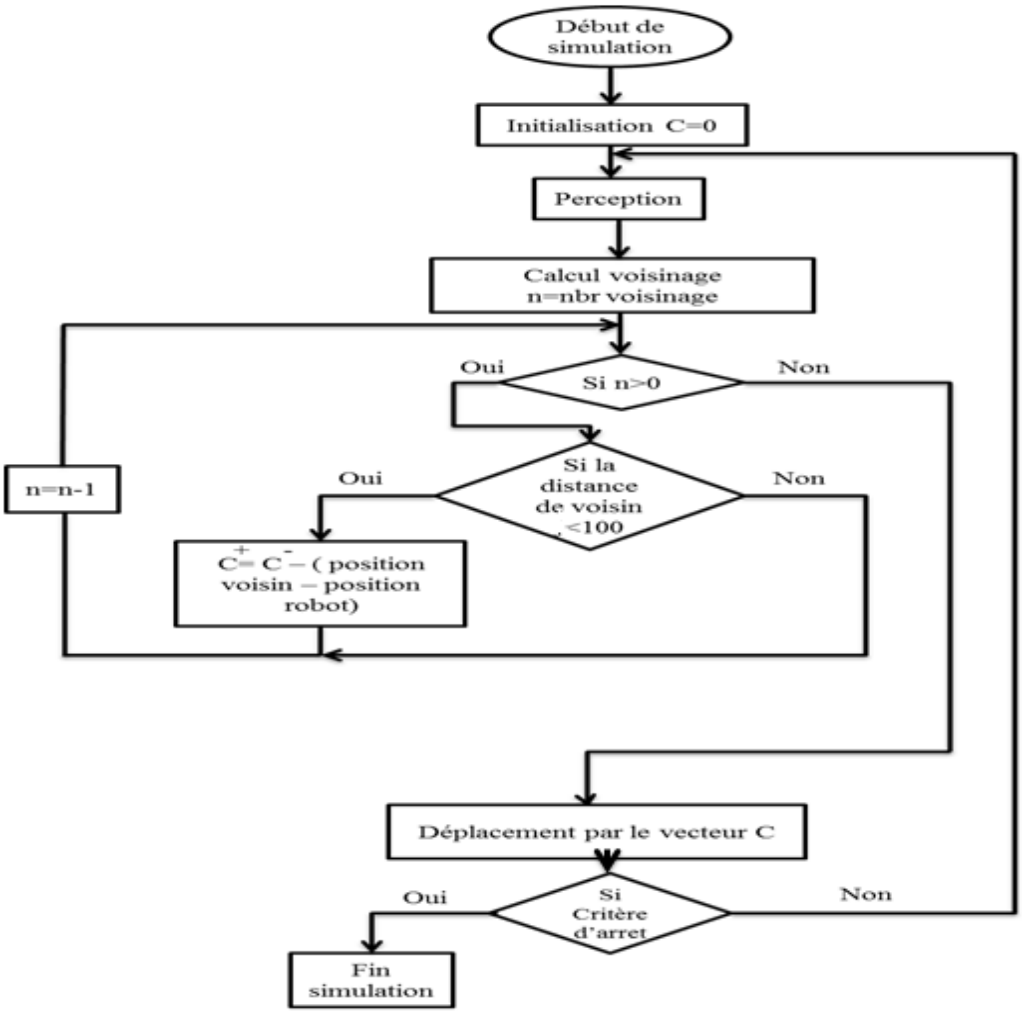


Figure 3- 3: Organigramme de séparation : auteur 2024

5. Alignement

5.1. Le pseudo code

Procédure règle 3(boid b_J)

Vecteur pv_J

Pour chaque boid b
Si b ! = b_J alors
$pv_J = pv_J + b.vitesse$
Fin si
Fin pour

$pv_J = pv_J / N-1$

Return $(pv_J – b_J.vitesse) / 8$

Fin procédure

5.2. Organigramme d'alignement

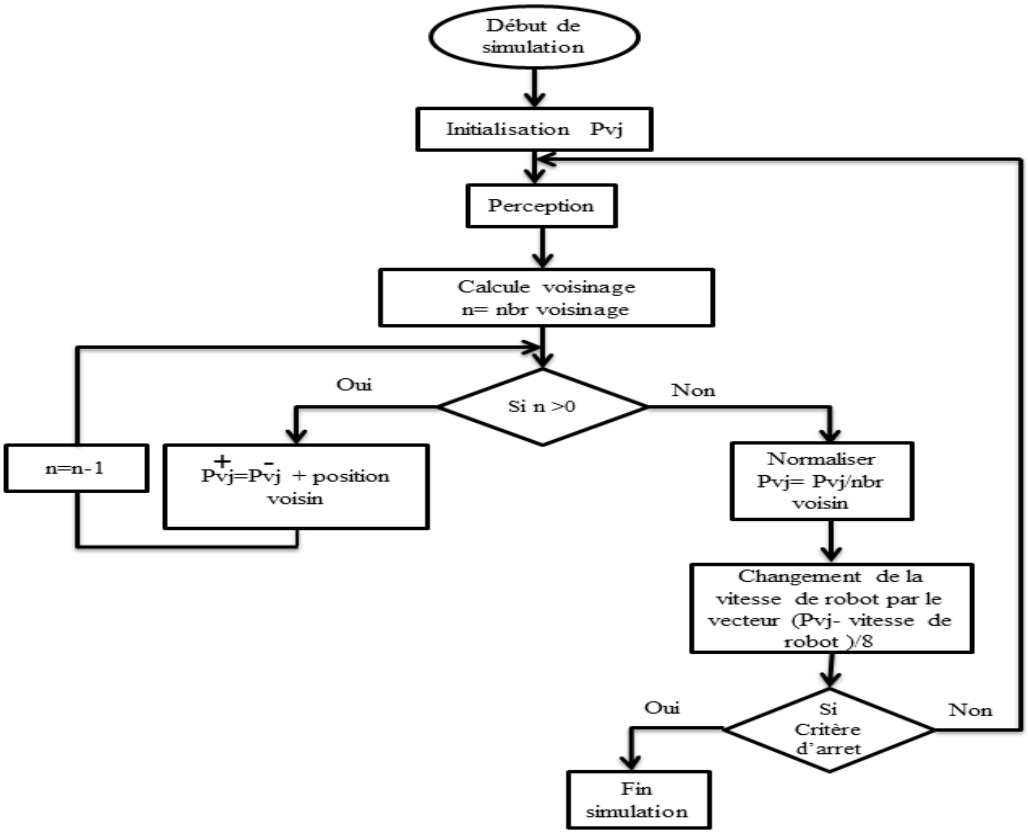


Figure 3- 4: Organigramme d'alignement : auteur 2024

6. Force attractive

6.1. Le pseudo code

Fonction calculerForceAttractive (position_actuelle, position_cible, ka) :

Force attractive = [0, 0]

Delta x = position cible [0] – position actuelle [0]

Delta y = position cible [1] – position actuelle [1]

Force attractive [0] = ka * delta x

Force attractive [1] = ka * delta y

Retourner force attractive

Fin fonction

6.2. Organigramme de force attractive

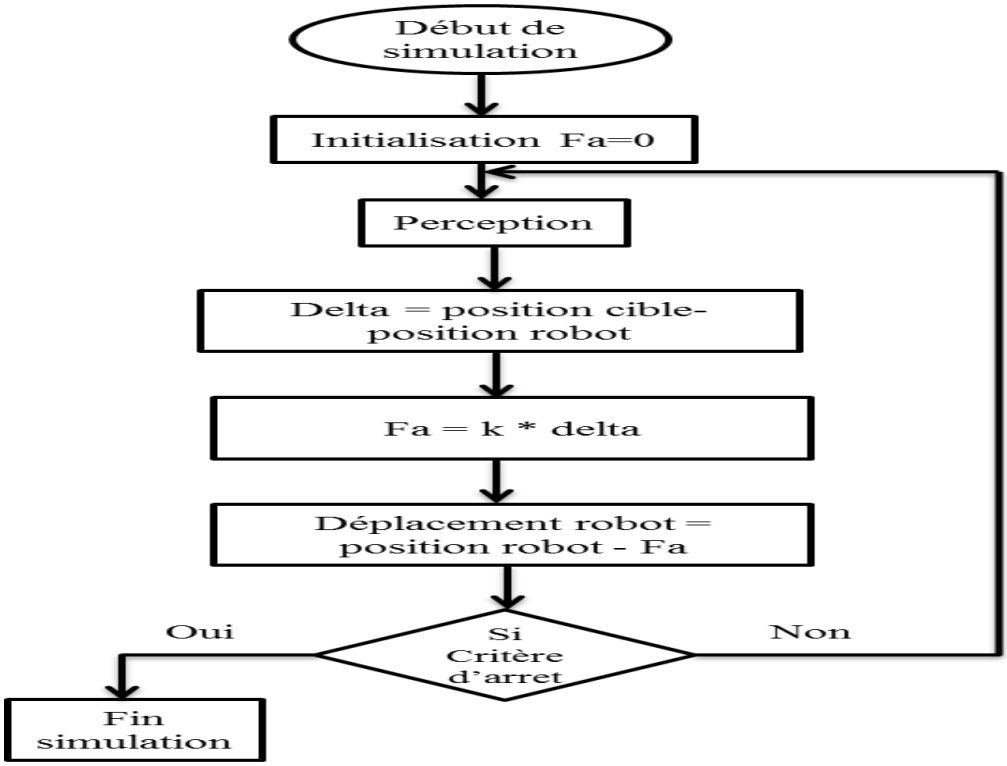


Figure 3- 5: Organigramme de force attractive : auteur 2024

7. Force répulsive

7.1. Le pseudo code

```
Fonction CalculerForceRepulsive (position_actuelle, obstacles) :
    Force répulsive = [0, 0]
    Pour chaque obstacle dans obstacles :
        Delta x = obstacle. Position x – position actuelle
        Delta y = obstacle. Position y – position actuelle
        Distance = sqrt (delta_x^2 + delta_y^2)
    Si distance < seuil :
            Force répulsive x = -1 * (delta x / distance)
            Force répulsive y = -1 * (delta y / distance)
            Force répulsive [0] += force répulsive x
            Force répulsive [1] += force répulsive y
    Fin si
    Fin pour
    Retourner force répulsive
Fin fonction
```

7.2. Organigramme de force répulsive

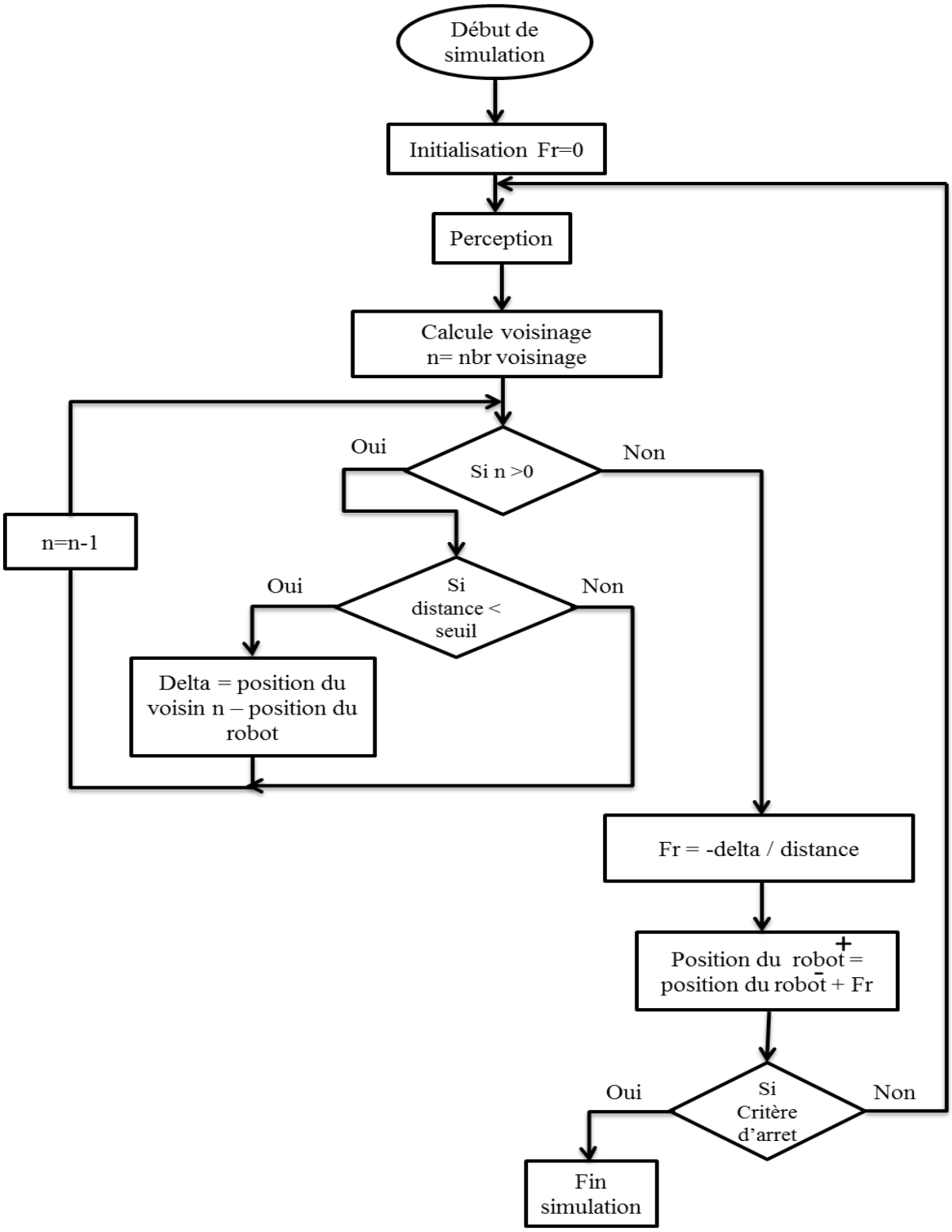


Figure 3- 6: Organigramme de force répulsive : auteur 2024

8. Graph de communication dans les systèmes ROS

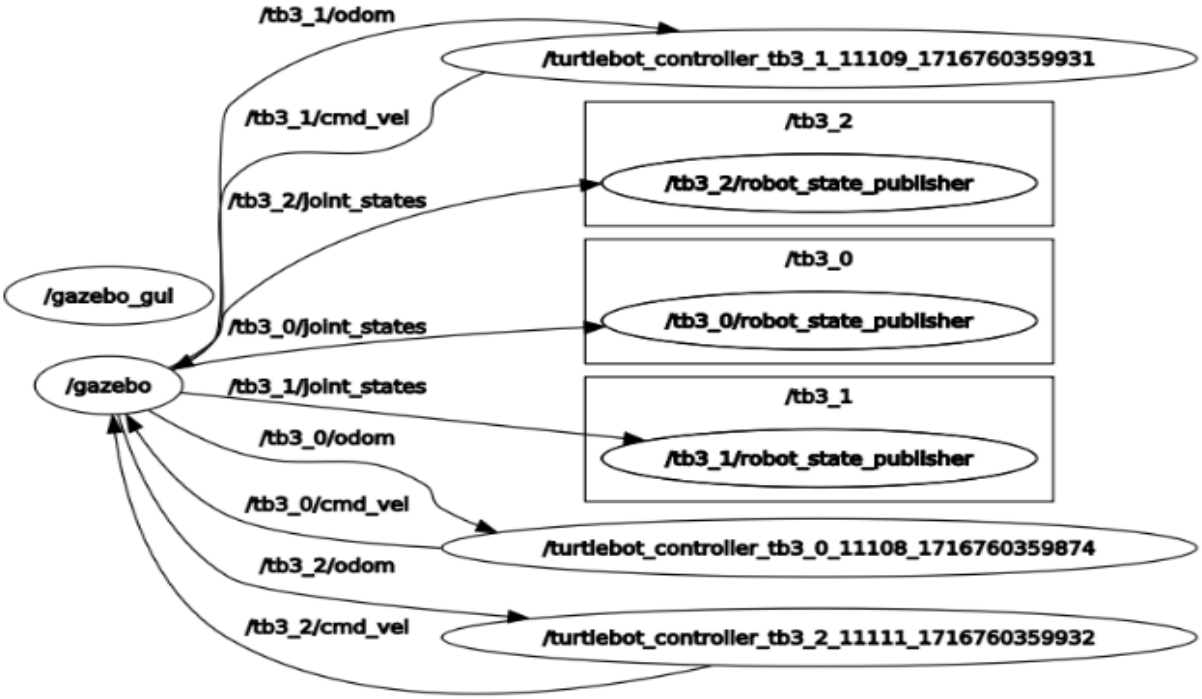


Figure 3- 7: Rqt-Graph

Ce graphe représente les interactions entre plusieurs composants d'un système de simulation robotique utilisant Gazebo et TurtleBot. Les nœuds de synchronisation des articulations (joint states) permettent de maintenir la cohérence entre les positions des différentes parties des robots. Les commandes de vitesse (cmd_vel) sont utilisées pour envoyer les vitesses linéaire et angulaire aux robots, ce qui leur permet de se déplacer. L'odométrie (odom) est essentielle pour récupérer la position des robots dans l'espace cartésien (x, y) ainsi que leur orientation dans l'espace d'Euler (yaw, pitch, roll). Ce schéma illustre comment les différents nœuds interagissent pour permettre une simulation cohérente et fonctionnelle des robots dans l'environnement virtuel de Gazebo.

9. Conclusion

Dans ce chapitre, nous avons détaillé les règles de cohésion, de séparation, d'alignement et d'attraction des objets. La conception a été validée à l'aide de schémas et pseudo codes décrivant le fonctionnement des règles et les différentes forces appliquées, ainsi que des schémas de communication ROS. Nous passons maintenant au chapitre suivant, qui abordera l'application pratique de notre système pour vérifier l'efficacité des concepts théoriques développés.

Chapitre 04 :

Implémentation et Résultats

1. Introduction

Après avoir fait une étude théorique on passe maintenant à la réalisation pratique. Dans le chapitre précédant on a établi une structure pour la conception de notre système, maintenant il ne reste plus qu'à passer à l'implémentation. Pour cela il nous faut tout d'abord établir une méthode à suivre pour la réalisation logicielle, par la suite nous procéderons aux tests et des scénarios ainsi qu'à leurs interprétations.

2. Environnement de travail

2.1. Visual studio code

Visual Studio Code est un éditeur de code source et un environnement de développement intégré (IDE) de Microsoft. Il est open-source et cross-Platform, c'est-à-dire qu'il fonctionne sur Windows, Linux et Mac. Il a été conçu pour les développeurs web, mais il prend en charge de nombreux autres langages de programmation tels que C++, C#, Python, Java, etc. Il offre de nombreuses fonctionnalités comme la coloration syntaxique, l'auto-complétion, la mise en évidence des erreurs, la navigation de code, le débogage, la gestion de versions, l'intégration avec Git, et beaucoup d'autres. Il est également extensible à l'aide d'une grande variété d'extensions développées par la communauté, permettant aux développeurs de personnaliser l'éditeur selon leurs besoins. [20]

2.2. Langage de programmation Python

Python est le langage de programmation open source le plus employé par les informaticiens.

Ce langage s'est propulsé en tête de la gestion d'infrastructure, d'analyse de données ou dans le domaine du développement de logiciels. En effet, parmi ses qualités, Python permet notamment aux développeurs de se concentrer sur ce qu'ils font plutôt que sur la manière dont ils le font. Il a libéré les développeurs des contraintes de formes qui occupaient leur temps avec les langages plus anciens. Ainsi, développer du code avec Python est plus rapide qu'avec d'autres langages. [21]

2.3. Pygame

Pygame est une combinaison de Python avec la SDL (Simple Direct média Library), une bibliothèque libre multi-plateformes permettant la gestion du multimédia dans la programmation.

- L'affichage vidéo 2D
- La gestion de l'audio
- La gestion de périphériques de commandes (clavier, souris…) [22]

2.4. Ros

Dans le domaine de la robotique, les plateformes revêtent une importance croissante. Une plateforme est divisée en une plateforme logicielle et une plateforme matérielle. La plateforme logicielle d'un robot contient des outils utilisés pour créer des programmes d'application pour robots, tels que la commande de dispositifs de bas niveau, SLAM (Simultaneous Localization and Mapping), la navigation, la manipulation, la reconnaissance d'objets ou d'humains, la détection et la gestion de paquets, le débogage et les outils de développement, en particulier dans l'industrie, au sein de l'entreprise. Les robots sont des outils de développement, en particulier dans l'industrie, où ils sont aujourd'hui principalement utilisés.

Les plates-formes matérielles pour robots étudient non seulement les plates-formes telles que les robots mobiles, les drones et les humanoïdes, mais aussi les produits commerciaux.

Par conséquent, les chercheurs en robotique du monde entier collaborent pour découvrir une plateforme qui soit intuitive et open source. La plateforme logicielle pour robots la plus populaire est ROS, qui signifie Robot Operating System.

ROS, le système d'exploitation des robots, est un cadre open source permettant de gérer les opérations, les tâches, les mouvements et autres activités des robots. ROS est destiné à servir de plateforme logicielle pour ceux qui construisent et utilisent des robots quotidiennement mais aussi pour ceux qui commencent à utiliser des robots depuis peu. Cette plateforme commune permet aux nouveaux venus d'être de plus en plus enclins à lire et elle est très facile à utiliser.

3 Cette structure de la plateforme permet d'utiliser le code et les informations partagées par les autres programmeurs, ce qui implique qu'il n'est pas nécessaire d'écrire tout le code pour déplacer les robots, pour cette raison, ROS a connu un succès remarquable. (Voir la figure 4-1).

Figure 4- 1 : Logo ROS [23]

2.4.1. Historique du ROS

Avant 2007, les premiers pas du projet ROS ont commencé à l'université de Stanford aux États Unis. Eric Berger et Keenan Wyrobek deux doctorants qui travaillent sur un projet de robotique, ils ont remarqué que leurs collègues multidisciplinaires trouvent des difficultés pour s'adapter avec d'autres domaines importants pour la robotique, d'où vient l'idée pour trouver une solution innovante pour résoudre ce problème. Pour décrire leur idée, Eric Berger a dit : « Quelque chose qui ne craignait pas, dans toutes ces différentes dimensions ». Quelques mois avant l'annonce officielle du ROS, la start-up a été incubé par « Willow Garage » ; une grande société des robots personnels et des services. Grâce au concept « open source », ROS Willow Garage a tenté de pénétrer le marché des robots de service commercial en 2013, mais a ni par se scinder en plusieurs start-ups, et a finalement été remis à l'Open Source Robotics Foundation (OSRF). La figure est met en ordre chronologique les versions du ROS.(Voir la figure 4-2). [23]

Figure 4- 2: Versions du ROS [23]

2.4.2. Les caractéristiques du ROS

Les principales caractéristiques de ROS peuvent être regroupées en cinq caractéristiques :

La première est la réutilisabilité du programme. Un utilisateur peut se concentrer sur l'objectif lié à l'application qu'il souhaite développer en téléchargeant le paquet correspondant pour les fonctions restantes. En même temps, il peut partager le programme qu'il a développé afin que d'autres puissent le réutiliser.

La deuxième caractéristique est que ROS est un programme basé sur la communication. Souvent, pour fournir un service, des programmes tels que des pilotes matériels pour les capteurs et les actionneurs et des fonctions telles que la détection, la reconnaissance et le fonctionnement sont développés dans un seul cadre.

Cependant, pour parvenir à la réutilisation du logiciel du robot, chaque programme et chaque fonction est divisé en petits morceaux basés sur sa fonction. C'est ce qu'on appelle la componentialisation ou modularisation en fonction de la plate-forme.

La troisième est le support des outils de développement. ROS fournit des outils de débogage, des outils de visualisation 2D (comme Rqt) et 3D (Rviz) qui peuvent être utilisés sans avoir à développer les outils nécessaires au développement du robot. Les outils qui facilitent de

visualiser l'état du robot et les performances des algorithmes, de déboguer les comportements défectueux.

Structure du système d'exploitation robotique Ros et d'enregistrer les données des capteurs. Il existe un rassemblement important et croissant d'algorithmes robotiques permettant de cartographier l'environnement, de s'y déplacer, de représenter et d'interpréter les données des capteurs, de planifier des mouvements, de manipuler des objets et d'effectuer d'autres opérations.

Par exemple, il y a de nombreuses occasions où il faut visualiser un modèle de robot tout en développant un robot. Le nombre croissant d'outils et leurs capacités permettent de vérifier directement le modèle du robot, mais aussi d'effectuer une simulation à l'aide du simulateur 3D fourni (Gazebo).

La quatrième est la communauté active. Ros est une communauté pour un logiciel open source open source. Il y a plus de 5 000 paquets qui ont été développés et partagés volontairement en 2017, les pages Wiki qui documentent de nombreux aspects du cadre, et un site de questions-réponses où vous pouvez demander de l'aide et partager ce que vous avez appris.

Le cinquième est la construction d'un écosystème. Diverses plateformes logicielles ont été développées et la plateforme la plus respectée et la plus utilisée d'entre elles, ROS (pour toutes les caractéristiques que nous avons déjà vues), est en train de façonner son écosystème. Elle crée un écosystème pour tout le monde : les développeurs de matériel dans le domaine de la robotique, comme les de capteurs, l'équipe opérationnelle de développement de ROS, les développeurs de logiciels d'application et les utilisateurs. [24]

2.4.3. Architecture et principe de fonctionnement

- **Notion de "méta operating system"**

ROS est l'abréviation de Robot Operating System, on peut donc dire sans risque qu'il s'agit d'un système d'exploitation. En particulier, ceux qui sont nouveaux à ROS pourraient penser qu'il s'agit d'un système d'exploitation similaire aux systèmes d'exploitation classiques. Cependant, une description plus précise serait que le ROS est un système de méta-exploitation. Bien que le terme anglais « Meta-Operating System » ne soit pas déni dans le dictionnaire, il décrit un système qui effectue des processus tels que la planification, le chargement, la surveillance et le traitement des erreurs en utilisant la couche de virtualisation entre les applications et les ressources informatiques distribuées. Par conséquent, le ROS fonctionne dans

un système d'exploitation existant tel qu'Ubuntu, qui est une des distributions de Linux et le plus recommandé par les experts. Après avoir terminé l'installation du ROS sur Ubuntu, les fonctionnalités fournies par le système d'exploitation tel que le système de gestion des processus, le système de fichiers, l'interface utilisateur et le programme l'utilité (compilateur, modèle de fil) peut être utilisée. En plus des fonctionnalités de base fournies par Ubuntu, Le ROS fournit les fonctions essentielles requises pour les programmes d'application des robots, comme les bibliothèques, la transmission/réception de données entre des matériels hétérogènes, l'ordonnancement et le traitement des erreurs. Ce type de logiciel est également appelé intergiciel ou cadre logiciel. [23]

2.4.4. Terminologie et composants du ROS

- **Maître « Master »**

Le maître agit comme un serveur de noms pour les connexions de nœud à nœud et la communication de messages. La commande « roscore » est utilisée pour exécuter le maître, et si vous exécutez le maître, vous pouvez enregistrer le nom de chaque nœud et obtenir des informations si nécessaire. La connexion entre les nœuds et la communication de messages tels que les « Topics » et les « services » –qui seront détaillés par la suite- sont impossibles sans le

Maître. Le maître communique avec les esclaves en utilisant XMLRPC (XML-Remote Procedure Call), qui est un protocole basé sur noeuds qui ne maintient pas la connectivité. En d'autres termes, les nœuds esclaves ne peuvent accéder que lorsqu'ils ont besoin d'enregistrer leurs propres informations ou de demander des informations à d'autres nœuds. L'état de connexion des uns et des autres n'est pas vérifié régulièrement. Grâce à cette caractéristique, le ROS peut être utilisé dans des environnements très vastes et complexes. XMLRPC est très léger et supporte une variété de langages de programmation, ce qui le rend bien adapté au ROS, qui supporte une variété de matériel et de langages de programmation. Lorsque vous exécutez un ROS, le maître sera configuré avec l'adresse URI et le port configuré dans le ROS_MASTER_URI. Par défaut, l'adresse URI utilise l'adresse IP de l'adresse locale PC, et le numéro de port 11311, sauf modification contraire.

- **Nœud « node »**

Un nœud fait référence à la plus petite unité de processeur fonctionnant en ROS. Il s'agit d'un programme exécutable. ROS recommande de créer un seul nœud pour chaque objectif, et il est recommandé pour une réutilisation facile. Par exemple, dans le cas des robots mobiles, le programme permettant de faire fonctionner le robot est décomposé en fonctions spécialisées. Un nœud spécialisé est utilisé pour chaque fonction, telle que l'entraînement des

capteurs, la conversion des données des capteurs, la reconnaissance des obstacles, l'entraînement des moteurs, l'entrée des encodeurs et la navigation. Au démarrage, un nœud enregistre des informations telles que le nom, le type de message, l'adresse URI et le numéro de port du nœud.

Le nœud enregistré peut agir en tant qu'éditeur, abonné, serveur de service ou client de service sur la base des informations enregistrées, et les noeuds peuvent échanger des messages en utilisant des sujets « Topics » et des services « services » (ces notions vont être détaillées dans la section 2.5.3) Le nœud utilise XMLRPC pour communiquer avec le maître et utilise XMLRPC ou TCPROS des protocoles TCP/IP pour communiquer entre les nœuds. La demande de connexion et la réponse entre les nœuds utilisent XMLRPC, et la communication des messages utilise TCPROS car il s'agit d'une communication directe entre les nœuds, indépendante du maître. Quant à l'adresse URI et au port numéro, une variable appelée ROS_HOSTNAME, qui est stockée sur l'ordinateur où se trouve le nœud est utilisée comme adresse URI, et le port est fixé à une valeur unique arbitraire.

- **Message**

Un nœud envoie ou reçoit des données entre les nœuds par l'intermédiaire d'un message. Les messages sont des variables telles que les nombres entiers, les nombres à virgule flottante et les booléens. Une structure de message imbriquée qui contient un autre message ou un ensemble de messages peut être utilisée dans le message. Le protocole de communication TCPROS et UDPROS est utilisé pour la livraison des messages.

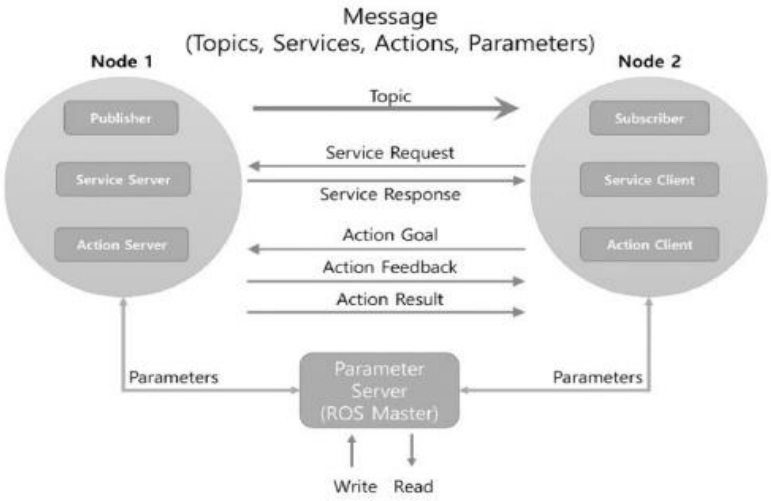


Figure 4- 3: Modes de communication [23]

- **Graph**

La relation entre les noeuds, les « Topics », les « publisher » et les « subscribers » présentée ci-dessus peut être visualisée sous forme de graphique. La représentation graphique de

la communication des messages n'inclut pas le service car il ne se produit qu'une seule fois. Le graphique peut être affiché en exécutant le nœud « rqt_graph » dans le paquet « rqt_graph ». Il existe deux commandes d'exécution, « rqt_graph » et « rosrun rqt_graph ». [23]

2.4.5. Modes de communication

Comme il est mentionné ROS est basé sur la communication entre les différentes noeuds, cette communication peut être établit en trois mode selon le besoin du développeur :

- **Topic**

Ce mode est asynchrone, il est basé sur la notion du « publisher » et « subscriber ». Comme les noms en Anglais l'indique, ce mode fonctionne comme suivant : Si les topics sont les routes permettant le transfert, la récupération, ou encore l'envoi de messages aux noeuds, alors les publishers et subscribers sont les bus transportant ces messages. C'est par leur biais que communiqueront les topics et noeuds. Ainsi, pour envoyer un message, un nœud utilisera un publisher pour transmettre son information au topic associé, s'il est possible de recevoir une information du composant relatif au nœud. Dans la même idée, un subscriber sera utilisé pour recevoir des informations venant d'un topic De plus, plusieurs noeuds, peuvent être des subscribers sur le même topic, voir la figure. Une fois la connexion est faite, le Topic transmit et reçoit les messages en continue, c'est pour cela, ce mode de communication est idéal pour les applications de mesure continue comme les radars.

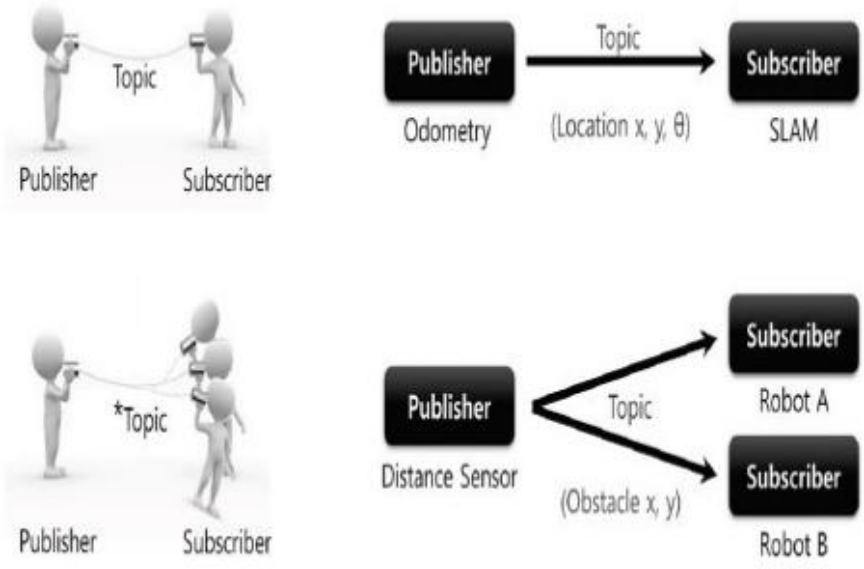


Figure 4- 4: Exemple de communication en mode Topic [23]

2.4.6. Outils et simulateurs ROS

ROS dispose de plusieurs outils qui peuvent être utiles lorsque le robot bouge ou qu'un algorithme est en cours d'exécution et que nous voulons comprendre s'il fonctionne correctement non. Il existe plusieurs outils ROS y compris ceux que les utilisateurs de ROS ont personnellement publiés. Les outils que nous allons décrire, qui représentent ceux qui ont été les plus utilisés au cours de expériences/tests en laboratoire sont les suivants : Rviz (outil de visualisation 3D), Rqt (qui est un cadre logiciel de ROS qui met en œuvre les différents outils d'interface graphique sous la forme de plugins), Rqt image-view (outil d'affichage d'images), Rqt graph (outil permettant de visualiser la corrélation entre les nœuds et les messages sous forme de graphique), Rqt plot et Gazebo, un simulateur 3D.

- **Rviz**

Rviz (Ros vizualisation) est l'outil de visualisation 3D de ROS. Le but principal est de montrer les messages ROS en 3D, ce qui nous permet de vérifier visuellement les données. Par exemple, il peut visualiser la distance entre le capteur (LDS) et un obstacle, les données de nuage des points (PCD) du capteur de distance 3D tel que RealSense, Kinect ou Xtion, la valeur de l'image obtenue à partir d'une caméra, et bien d'autres choses encore, sans avoir besoin de développer le logiciel séparément.

Figure 4- 5: Logo Rviz [23]

- **Gazebo**

Gazebo offre la possibilité de simuler avec précision et efficacité des populations de robots dans des environnements intérieurs et extérieurs complexes. Au bout de vos doigts, vous disposez d'un moteur physique robuste, de graphiques de haute qualité et d'interfaces graphiques et programmatiques pratiques. Mieux encore, Gazebo est gratuit avec une communauté dynamique (source site o-ciel gazebosim.com).

Ces caractéristiques sont :

- Simulation de la dynamique
- Graphiques 3D avancés
- Capteurs et bruit
- Plugins
- Modèles de robots
- Transport TCP/IP
- Simulation de nuages
- Outils en ligne de commande

Figure 4- 6: Logo Gazebo [23]

- **Rqt**

Outre l'outil de visualisation 3D Rviz, ROS fournit divers outils d'interface graphique pour le développement des robots. Par exemple, il existe un outil graphique qui montre la hiérarchie de chaque nœud sous forme de diagramme, indiquant ainsi l'état du nœud et du « topic » en cours, et un outil de traçage qui schématise un message sous forme de graphique en 2D. À partir de la Version ROS Fureté, plus de 30 outils de développement d'interface graphique ont été intégrés en tant qu'outil appelé rqt qui peut être utilisé comme un outil GUI complet. [23]

3. Configuration du matériel utilisé

Nous avons travaillé sur deux PC des marques Hp et Asus avec des propriétés suivants :

Pc	Asus	Hp
Ram	4,00 Go	4,00 Go
Processeur	Intel I7 4500U 1.80 Ghz	Intel I3 7020U 2.3GHZ
Logiciel	Ubuntu 20.04	

4. Scénarios de tests :

Scénario 1

Cet environnement contient 2 obstacles et une cible :

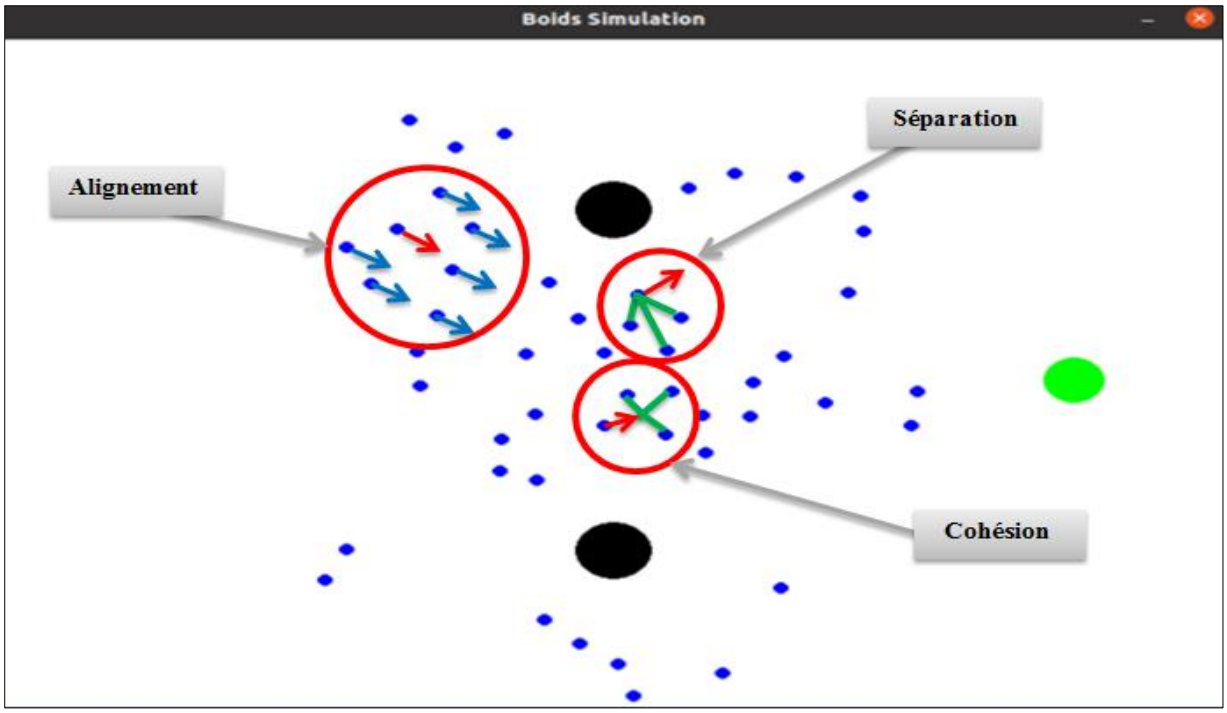
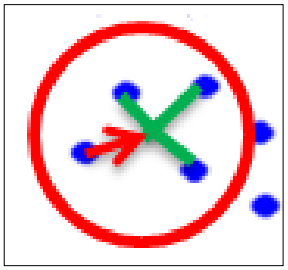


Figure 4- 7: Application des règles de boids

Scénario 1 : Cohésion

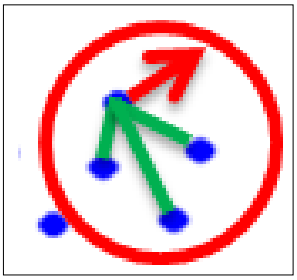


Cette image représente comment les boids utilisent la règle de cohésion pour rester groupés en se rapprochant les uns des autres. La force résultante indiquée par la flèche rouge est le vecteur de déplacement que le boid central doit suivre pour se conformer à cette règle.

Ci-dessus on donne le tableau représentent la cohésion :

Boid central	**Boids voisin**	**Pc_j**	**Force résultante de cohésion**
R(50,50)	B1(40,60)	(40,60)	(0.05, 0.0167)
	B2(70,50)	(110,110)	
	B3(55,45)	(165,155)	

Scénario 1 : Séparation

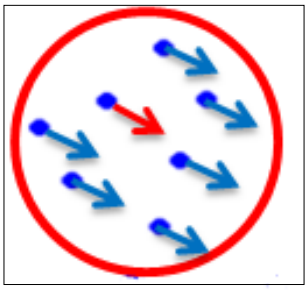


Cette image représente comment les boids utilisent la règle de séparation pour éviter les collisions en maintenant une distance appropriée les uns des autres. La force résultante indiquée par la flèche rouge est le vecteur de déplacement que le boid central doit suivre pour se conformer à cette règle.

Ci-dessus on donne le tableau représentent la séparation :

Boid central	**Boids voisin**	**C**	**Force résultante de séparation**
R(50,50)	B1(40,60)	(10,-10)	(-15, -5)
	B2(70,50)	(-10,-10)	
	B3(55,45)	(-15,-5)	

Scénario 1 : Alignement



Cette image représente comment les boids utilisent la règle d'alignement pour rester dans la même direction en s'alignant les uns avec les autres. La flèche rouge indique le vecteur de déplacement que le boid central doit suivre pour se conformer à cette règle.

Ci-dessus on donne le tableau représentent l'alignement :

Boid central	Boids voisin	Pv_j	Force résultante d'alignement
R(2,3)	B1(3,4)	(3,4)	(0.084, 0.084)
	B2(4,2)	(7,6)	
	B3(1,5)	(8,11)	

Scénario 2

Il existe une répulsion entre les obstacles et les robot.

Concernant la cible contient une charge négative et donc attraction. :

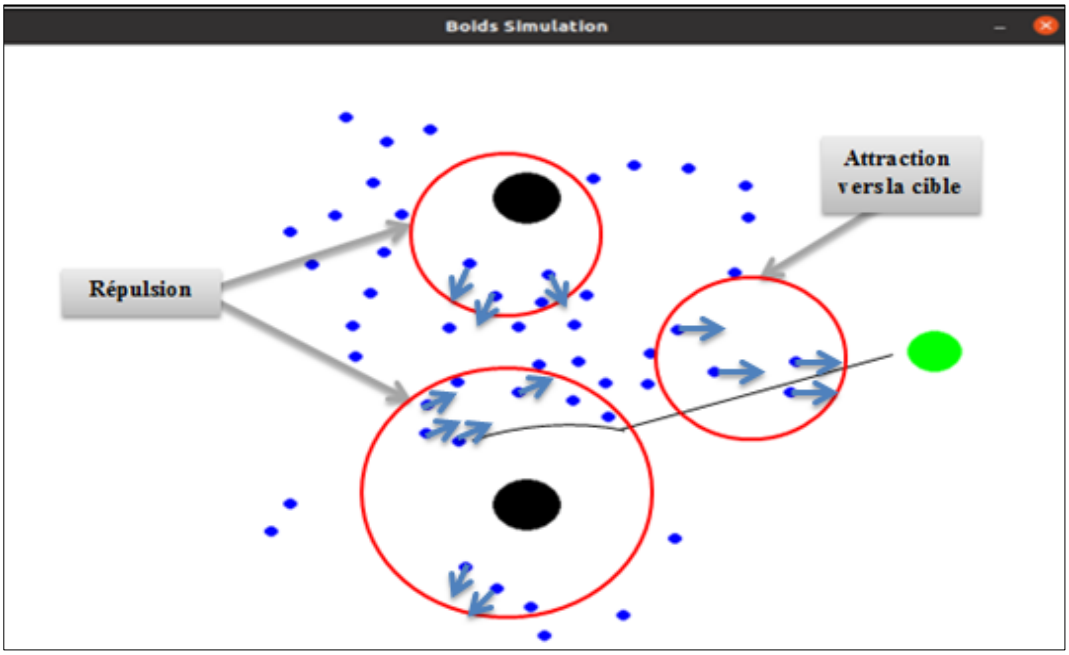
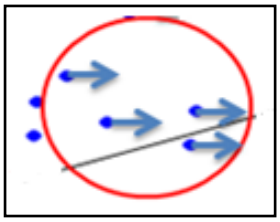


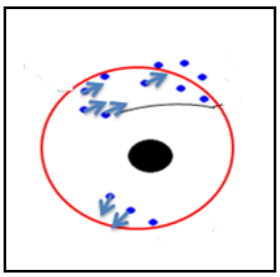
Figure 4- 8: Application des règles de champ de potentiel

Scénario 2 : Attraction



Cette image montre que les boids sont attirés vers la cible, ce qui indique qu'une force d'attraction les dirige.

Scénario 2 : Répulsion



Cette image montre que les boids sont repoussés loin de l'obstacle, ce qui indique qu'une force de répulsion les éloigne.

Scénario 3

Cet environnement contient plusieurs obstacles et une cible :

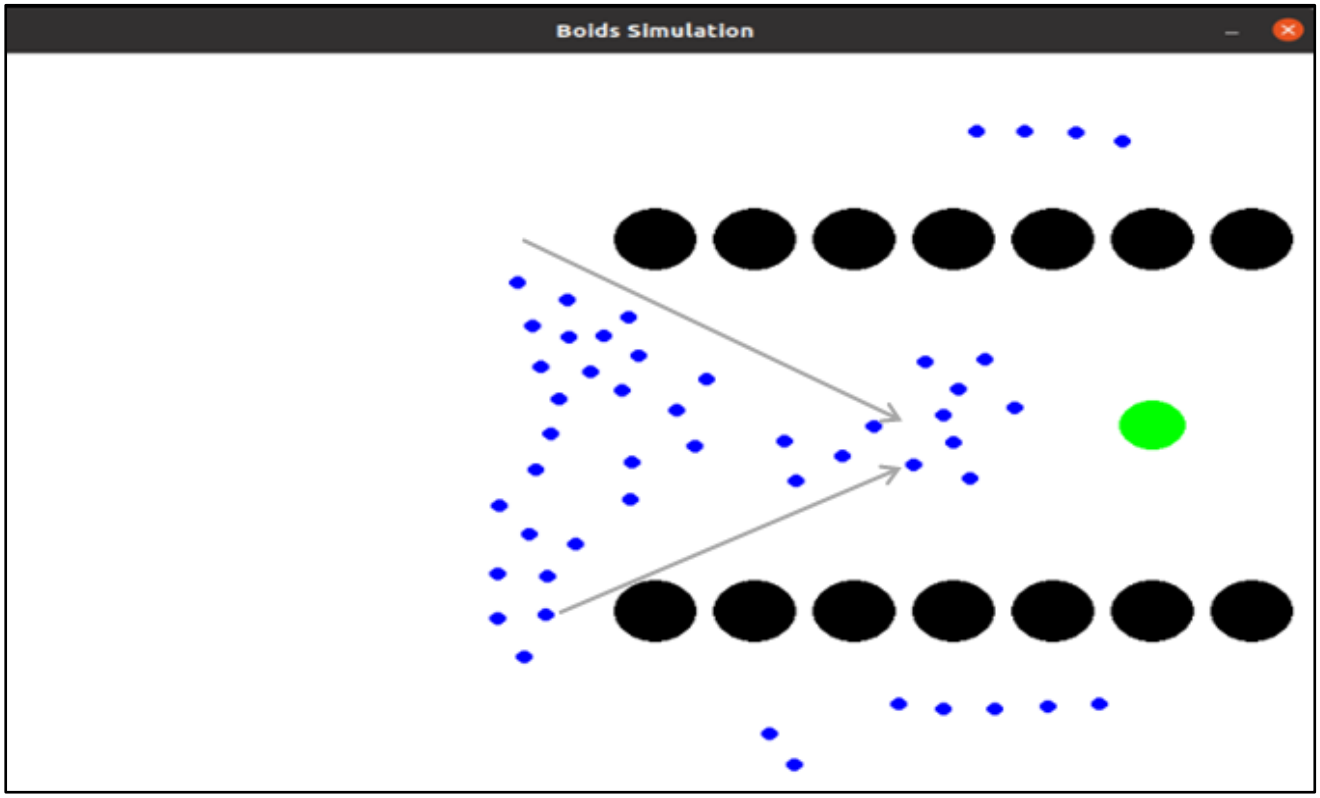


Figure 4- 9: Navigation dans un environnement plein d'obstacles

Scénario 4 : test dans l'environnement Gazebo

Pour cette conception d'environnement adéquat nous avons opté pour une simulation Gazebo de forme rectangulaire et dont les caractéristiques sont les suivantes taille 100 x 100 (u) couleur gris. Entre autres on distingue la forme cylindre pour les obstacles d'une couleur gris foncé.

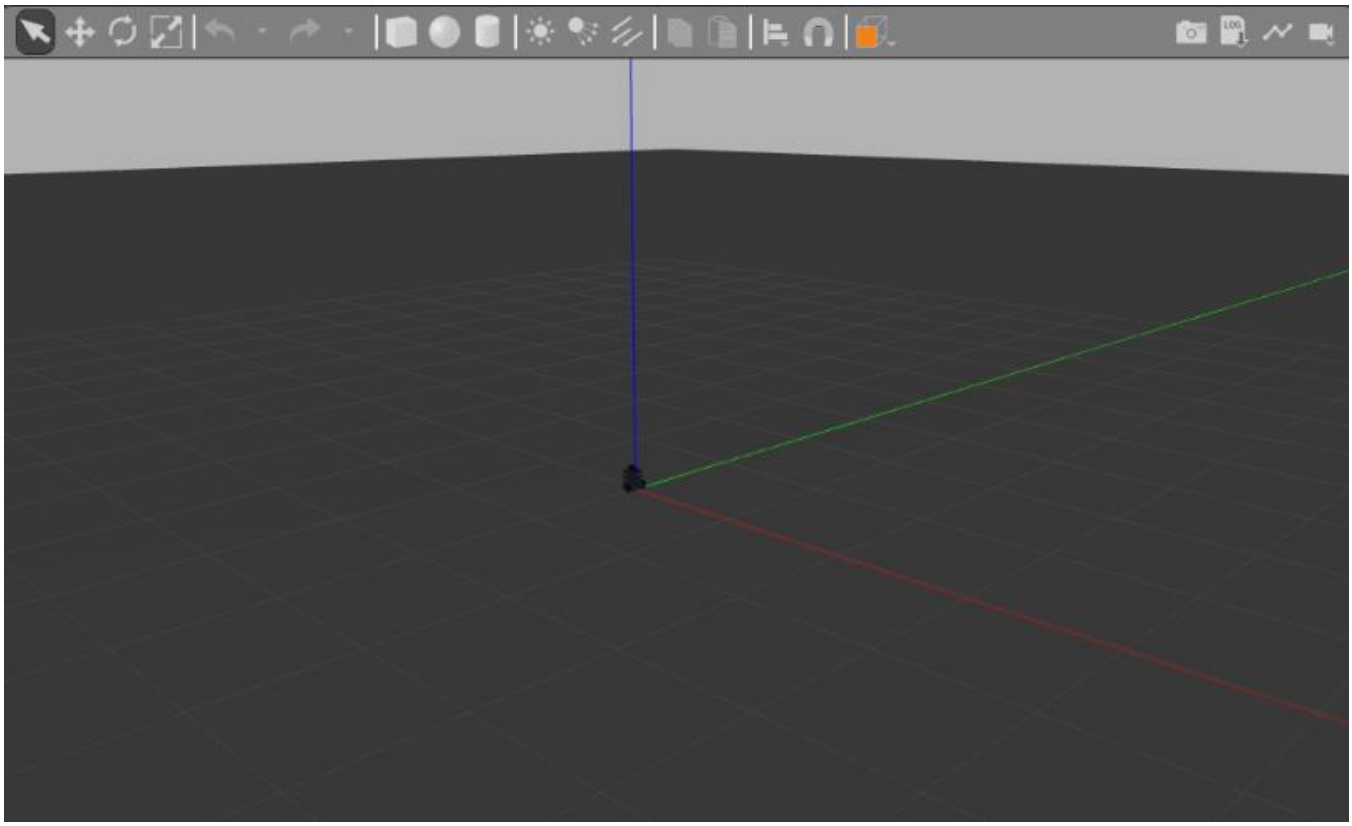


Figure 4- 10: Environnement vide

Environnement plain d'obstacles :

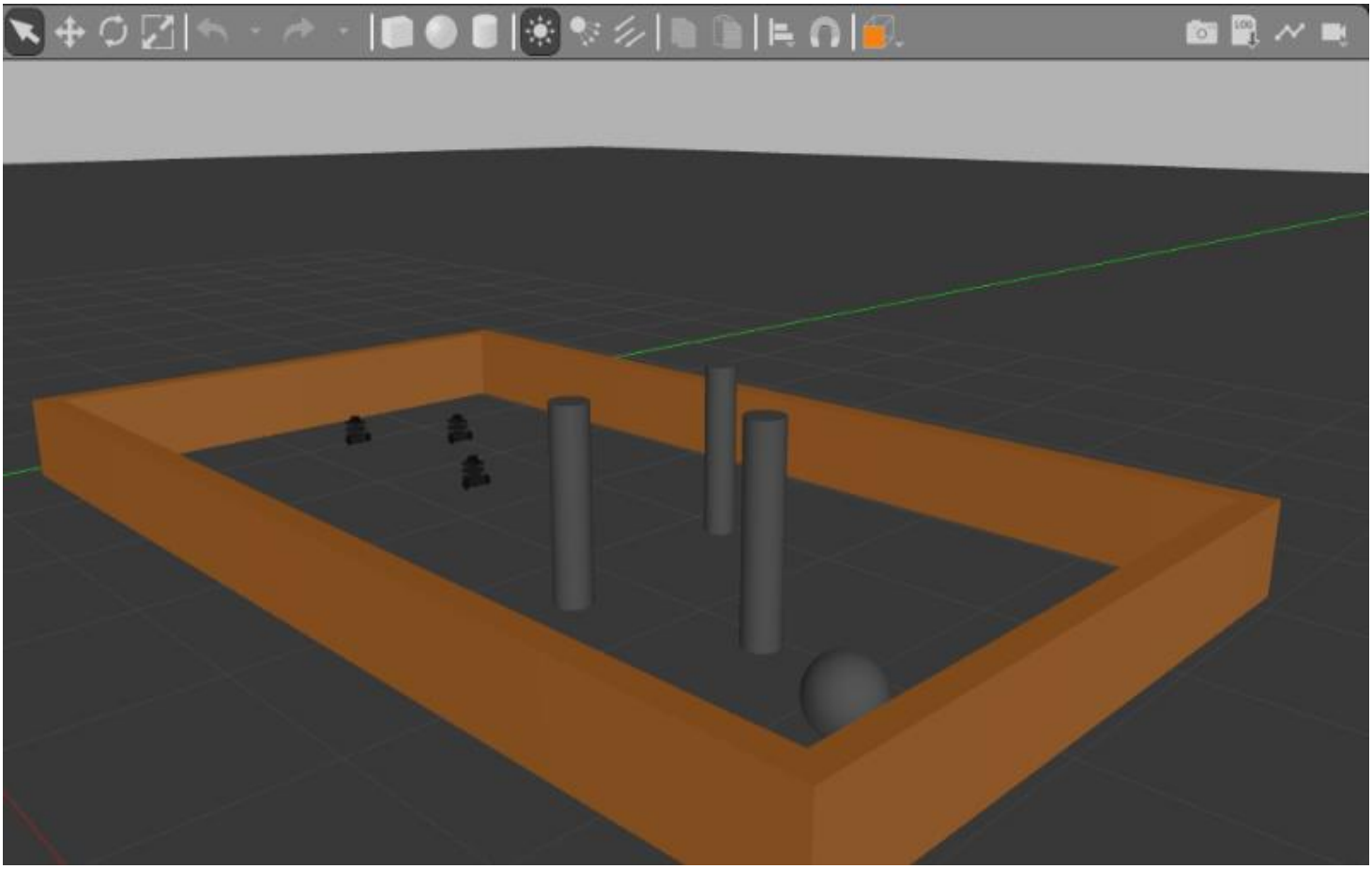


Figure 4- 11: Navigation dans un environnement plain d'obstacles.

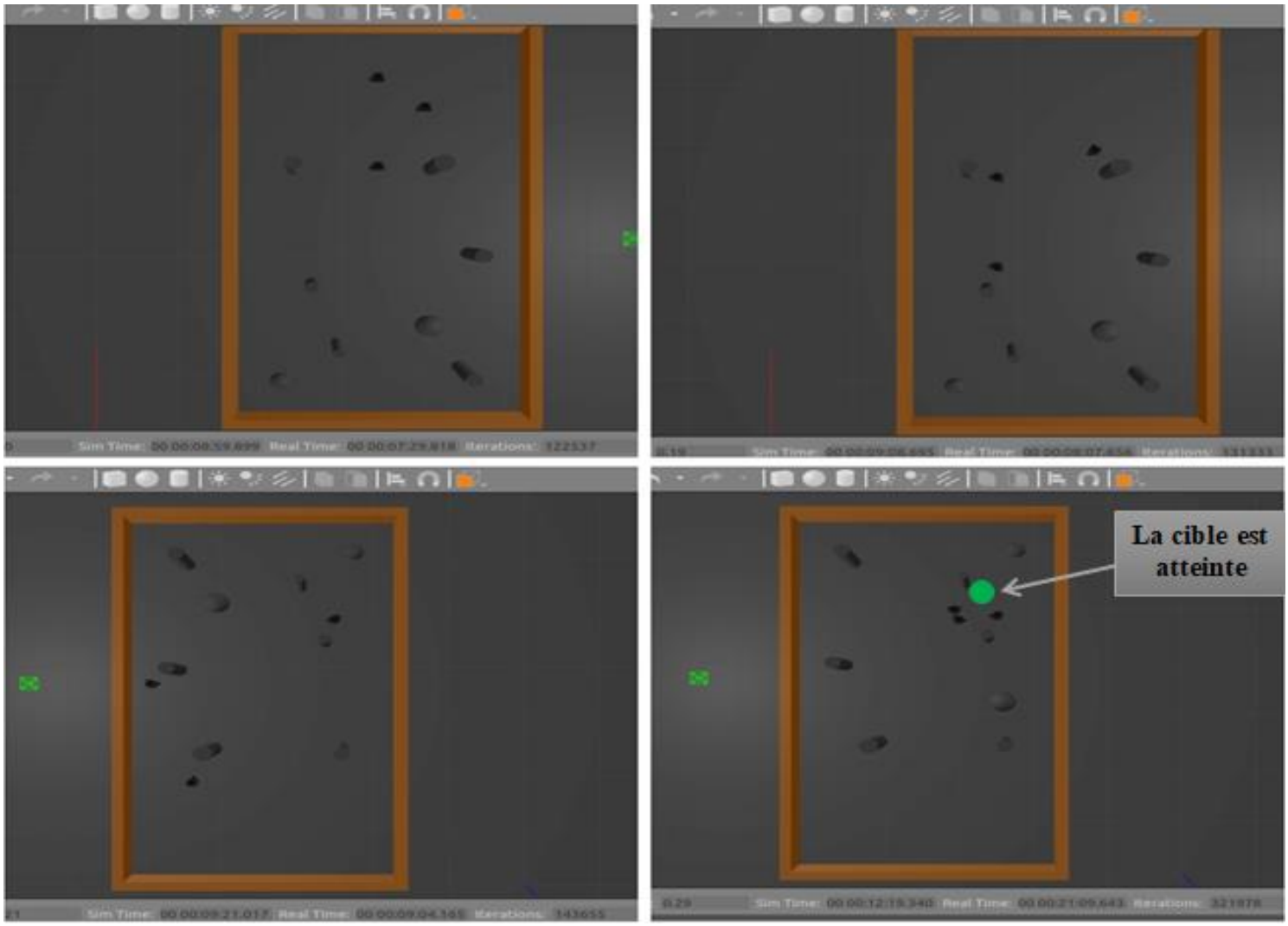


Figure 4- 12: Les étapes de navigation multi-robot dans un environnement plain

Itération 1 **Itération finale**

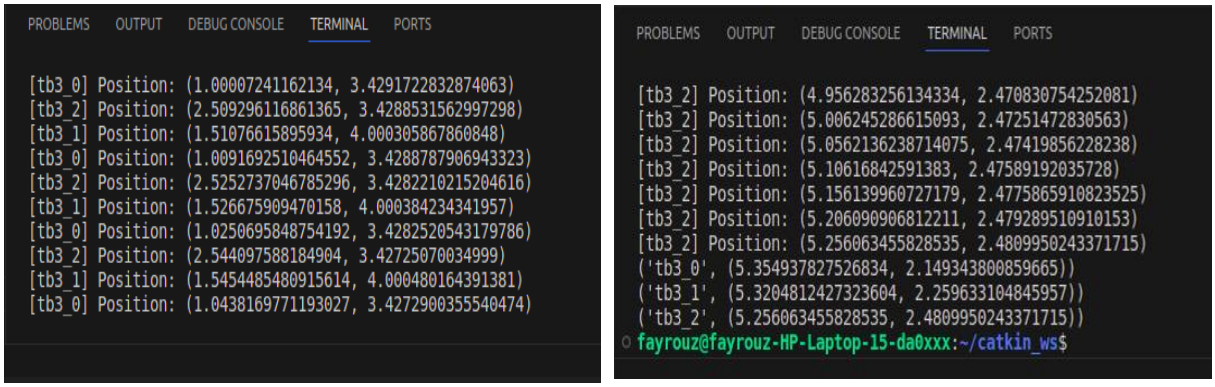


Figure 4- 13: Montre la convergence des turtlebots vers la cible

5. Analyse de résultat

5.1. Statistique sur la distance moyenne entre les boids et la cible

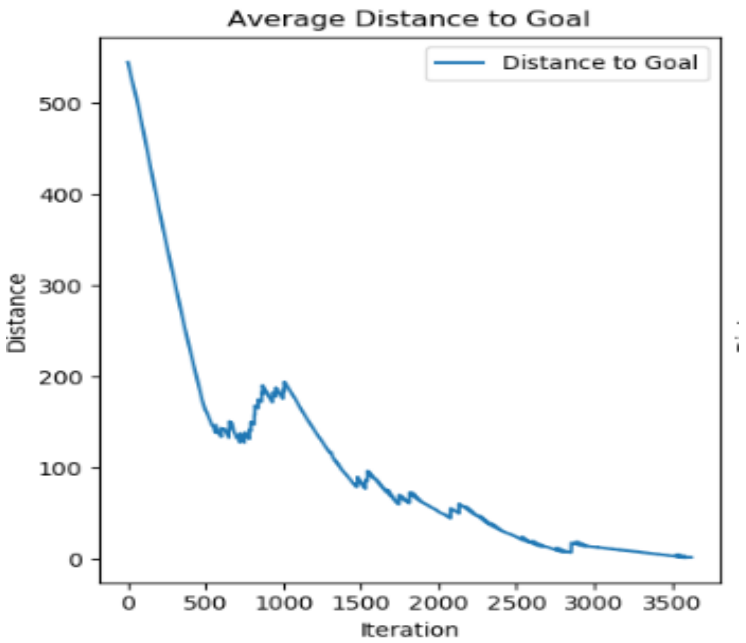


Figure 4- 14: Distance moyenne entre les boids et la cible

Le graphique montre que la distance moyenne entre les boids et la cible se décroit avec le temps.

5.2. Statistique sur la distance de séparation inter boids

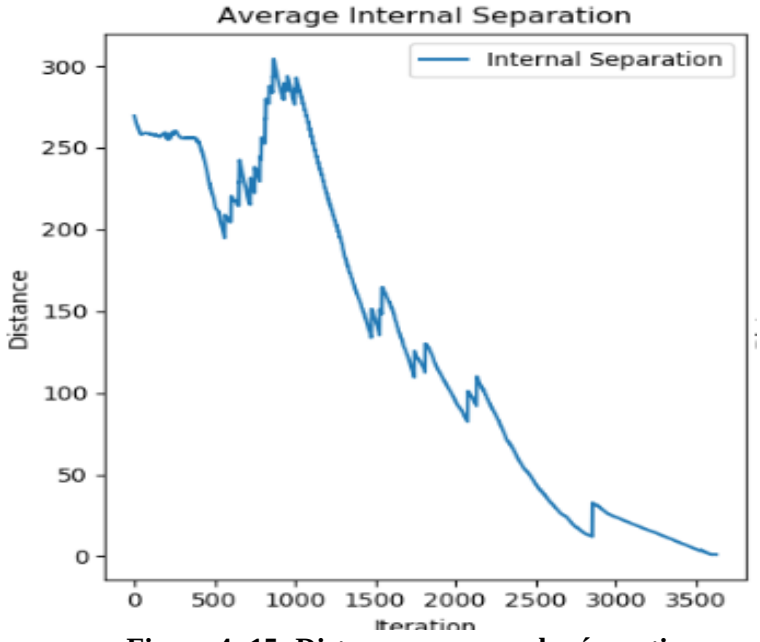


Figure 4- 15: Distance moyenne de séparation

Le graphique montre que l'écart entre les boids diminue progressivement avec le temps, ce qui indique que les boids se rassemblent de plus en plus.

5.3. Statistique sur l'alignement (la divergence des directions des boids)

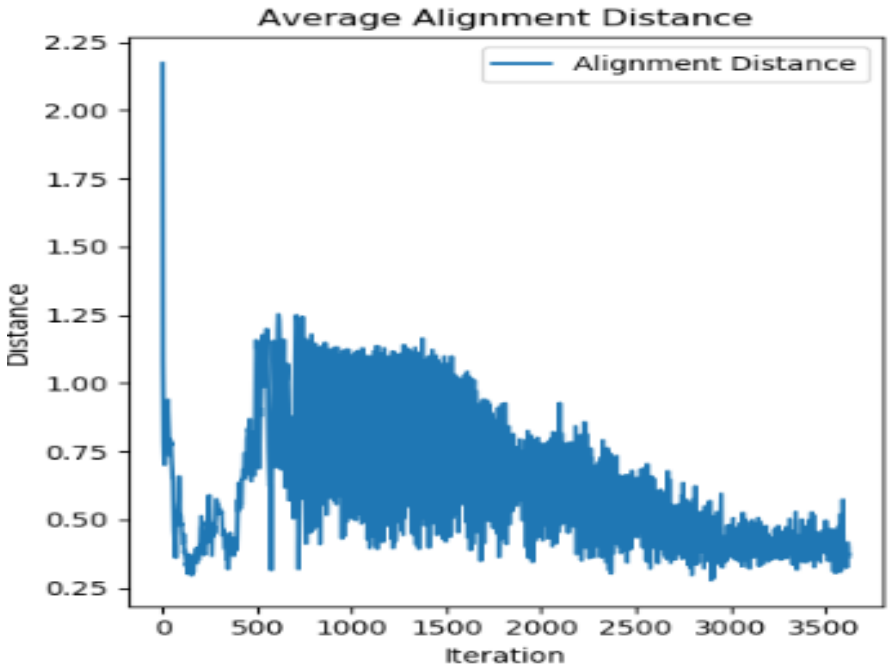


Figure 4- 16: Divergence moyenne d'alignement

Le graphique montre que la divergence entre les boids et l'objectif diminue progressivement avec le temps, ce qui indique que les boids suivent une même direction.

6. Conclusion

Le Chapitre a abordé l'implémentation de notre système multi-robots. Nous avons utilisé des outils tels que Visual Studio Code pour le développement, Python comme langage de programmation, Pygame pour les simulations de base, et Gazebo pour des environnements de simulation plus complexes. Les résultats des tests dans des environnements simulés ont démontré l'efficacité de notre approche pour la navigation et l'évitement des obstacles. Les défis rencontrés, comme les minima locaux dans les champs de force, ont été analysés et des solutions ont été proposées.

Conclusion générale

Conclusion générale

En conclusion, la navigation robotique autonome dans des environnements complexes reste un défi majeur qui nécessite des méthodes robustes et adaptables. Notre étude a montré que la combinaison de la méthode du champ potentiel et du modèle primitif de Reynolds permet aux robots de naviguer de manière autonome et sûre tout en conservant une configuration fixe.

L'utilisation d'outils tels que Pygame et Gazebo à différentes étapes de mise en œuvre a été cruciale pour mettre en œuvre et valider nos concepts théoriques. Cependant, la mise en œuvre a révélé quelques problèmes, notamment le minimum local dans les environnements complexes.

Pour surmonter les défis auxquels nous sommes confrontés dans des scénarios plus complexes, nous proposons d'ajouter des techniques de planification (comme RRT) pour éviter les problèmes de minimum local afin de la rendre plus robuste et adaptable. Les travaux futurs devraient se concentrer sur ces améliorations pour gérer des environnements plus dynamiques et imprévisibles, garantissant une navigation fiable et efficace des robots dans différentes situations.

Bibliographie

Bibliographie

[1] Dernier Accès le 25/05/2024, Récupéré sur : https://fr.wikipedia.org/wiki/Robotique .

[2] https://biblio.univ-annaba.dz/ingeniorat/wp-content/uploads/2019/09/Bouchaib-Hamza.pdf .

[3] Dernier Accès le 20/05/2024, Récupéré sur : https://vitrinelinguistique.oqlf.gouv.qc.ca/fiche-gdt/fiche/8383175/systeme-multiagent

[4] Bernard BAYLE. "robotique mobile",livre,Ecole Nationale Supérieure de Physique de Strasbourg Université de Strasbourg, 2005.

[5] J.Sawssen. "Optimisation de la navigation robotique ". Thèse Doctorat, Université de Toulouse, 2016.

[6] MERABTI. Hocine. "Approches bio-inspirées pour la reconnaissance de formes". ²Thèse Doctorat, Université 8 Mai 1945 Guelma.

[7] Olivier Lefebvre. "Navigation autonome sans collision pour robots mobiles nonholonomes". Thèse Doctorat, Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

[8] Dernier Accès le 02/06/2024, Récupéré sur : https://www.lemagit.fr/definition/logiquefloue.

[9] Miha Moskon, Miha Mraz, Nikolaj Zimic, Iztok Lebar Bajec. "FUZZY MODEL OF BIRD FLOCK FORAGING BEHAVIOR". University of Ljubljana, Faculty of Computer and Information Science 1000 Ljubljana, Trza ˇ ska 25, Slovenia.

[10] Alexandre Bonnefond, Olivier Simonin. "Extension des Modèles de Flocking aux Environnements avec Obstacles et Communications Dégradées". ; Univ. Lyon, Inria, INSA de Lyon, CITI & LIP Labs, 6 Av. des Arts 69621 Villeurbanne cedex (France).

[11] Dernier Accès le 09/06/2024,, Récupéré sur : https://www.sage.com/fr-fr/blog/glossaire/intelligence-artificielle-ia-definition/(intelligence) .

[12] Fredy Mart´ınez, Holman Montiel, Luis Wanumen. " A deep reinforcement learning strategy for autonomous robot flocking " . Facultad Tecnologica, Universidad Distrital Francisco Jos ´ e de Caldas, Bogot ´ a D.C, Colombia.

[13] Andreas Brandstatter, Radu Grosu. " multi-Agent Spatial Predictive Control with Application to Drone Flocking (Extended Version). Technische Universitat Wien (TU

Wien), Austria, Department of Computer Science, Stony Brook University, USA.

[14] Dernier Accès le 12/06/2024, Récupéré sur : https://www.lemondedupc.fr/article/98-les-boids-quand-l-informatique-imite-la-nature.

[15] Dernier Accès le 11/06/2024, Récupéré sur : https://medium.com/nerd-for-tech/local-path-planning-using-virtual-potential-field-inpython-ec0998f490af

[16] Dernier Accès le 15/06/2024, Récupéré sur : https://code.tutsplus.com/3-simple-rules-of-flocking-behaviors-alignment-cohesion-and-separation--gamedev-3444t

[17] Dernier Accès le 07/06/2024, Récupéré sur : http://www.univ-usto.dz/theses_en_ligne/doc_num.php?explnum_id=1423.

[18] Aslan Devbrat, "Local Path Planning Using virtual Potential Field in Python".

[19] Jiubo Sun, Guoliang Liu, Guohui Tian, and Jianhua Zhang. "Smart Obstacle Avoidance Using a Danger Index for a Dynamic Environment". School of Control Science and Engineering, Shandong University, Jinan 250061, China.

[20] Dernier Accès le 17/06/2024, Récupéré sur : https://bility.fr/definition-visual-studio-code.

[21] Dernier Accès le 17/06/2024, Récupéré sur : https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique.

[22] Dernier Accès le 10/06/2024, Récupéré sur : https://info.blaisepascal.fr/pygame.

[23] Imed-Eddine GUEBLI. "Navigation et commande d'un robot mobile à 4 roues sous ROS". Diplôme de MASTER, Université 8 Mai 1945 Guelma.

[24] Morgan Quigley, Brian Gerkey, and William D.Smart, Programming Robots with ROS, December 2015.

[25] Dernier Accès le 12/06/2024, Récupéré sur: https://fr.123rf.com/photo_61245587_photographi%C3%A9-close-up-ciel-bleu-dans-lequel-une-bande-d-oiseaux-voler-silhouettes-visibles-de.html.

[26] Dernier Accès le 16/06/2024, Récupéré sur: https://medium.com/@sebastian.barros/nature-simulation-systems-for-emergent-gameplay-a2207cce6ac4.

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليـــــــــم العالـــــــــي و البحث العلمـــــــــي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جـامعة 20 أوت 1955- سكيكدة
كـــــلية العلـــــوم
مكتبـة الكليـة

Université 20 Août 1955 - Skikda
Faculté des Sciences
Bibliothèque de la faculté

سكيكدة في / /2024

بطاقة معلومات خاصة بمذكرة التخرج

اسم و لقب الطالب : رقم التسجيل :

* ساعد حميد ثني مبروز ..... * 360056 15
* رماش ني هالة ..... * 360094 59
* ..... * .....
* ..... * .....

اسم و لقب المشرف على المذكرة : لو الطيب رشيد

عنوان المذكرة : Système de navigation multi-robots inspiré des vols d'oiseaux

القسم : إعلام آلي

المستوى : M2

التخصص : R.S.D

الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

وزارة التعليـــــــــم العالـــــــــي و البحث العلمـــــــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université 20 Août 1955- skikda-

Faculté des Sciences

Département d'Informatique

جـامعة 20 أوت 1955- سكيكدة

كـــــلية العلـــــوم

قسم الاعلام الالي

الرقم : ..../ ق/ أ/ ك.ع/2024

Autorisation de Dépôt de Mémoire de Master

Je soussigné: ..Bantine Rachid.....

Certifie que l'étudiant(e) :...Saâd hamidache feyrouz.....

Spécialité : ...Réseaux et systèmes distribués.....

Ayant soutenu le projet intitulé :..Système de navigation multi-robots inspiré des volées d'oiseaux.....
.....

A apporté les corrections nécessaires sur son manuscrit de Master

Signature de l'encadreur

الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

وزارة التعليـــــــــم العالـــــــــي و البحث العلمـــــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université 20 Août 1955- skikda-

Faculté des Sciences

Département d'Informatique

جـامعة 20 أوت 1955- سكيكدة

كـــــلية العلـــــوم

قسم الاعلام الالي

الرقم: ..../ ق/ ا/ ا/ ت ع/2024

Autorisation de Dépôt de Mémoire de Master

Je soussigné: Boutine Rachid

Certifie que l'étudiant(e) : Remmache Hala

Spécialité : Réseaux et systèmes distribués

Ayant soutenu le projet intitulé : Système de navigation multi-robots inspiré des volées d'oiseaux

A apporté les corrections nécessaires sur son manuscrit de Master

Signature de l'encadreur

Rachid