REPUBLIQUE ALGERIENNE DEMOQRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Khider – BISKRA

Faculté des sciences Exactes, des Sciences de la Nature et de la vie

Département d'informatique

N° d'ordre :IVA 05/IVA/M2/2023

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Image et vie Artificielle (IVA)**

Etude comparative des algorithmes de la navigation globale des robots mobiles

---

Par :

AZZAOUI SOUMIA

Soutenu le 20/06/2023 devant le jury composé de :

Zerari Abd El Mounène	MCB	Président
Cherif Foudil	Professeur	Rapporteur
Benchabane Moufida	MAA	Examinateur

Année universitaire 2022-2023

Dédicaces:

Je dédie ce travail :

A mes parents, mon père Boudjemaa et ma mère Sarah pour leur patience et leurs encouragements à mon égard en tout temps et pour leurs conseils.

A mes frères, de l'aîné d'entre eux, mon frère Muhammad, au plus jeune frère de la famille, Salah al- Din.

A mes chères amies : Asma, Rafika, Fatima, Rima, Amal, Ikram, Sarah, Manal, Rahma...etc.

Remerciements:

Tout d'abord, je remercie Dieu Tout-Puissant, grâce à qui je suis arrivé ici, et qui m'a donné la patience et le désir de faire ce travail. Dieu soit loué toujours et à jamais.

Je tiens à exprimer mes sincères remerciements au superviseur, Foudil Cherif, pour son aide, ses conseils et ses encouragements. Je tiens également à remercier les membres du jury d'avoir accepté l'évaluation de mon travail, ainsi que ma famille, ma mère, mon père et mes frères pour leur soutien dans les moments difficiles.

Table de matières

Liste des figures

Liste des figures

Liste des tableaux

Résumé

L’objectif de notre étude vise à analyser ces trois algorithmes (Dijkstra, RRT, A*) dans le contexte de la navigation globale des robots autonomes, qui consiste à planifier un itinéraire complet d'un point de départ à un point d'arrivée tout en évitant les obstacles. L'algorithme A* est largement utilisé pour sa capacité à trouver efficacement le chemin le plus court, en utilisant une heuristique pour guider la recherche. L'algorithme de Dijkstra, quant à lui, trouve le chemin le plus court en explorant tous les nœuds du graphe, sans utiliser d'heuristique. Enfin, l'algorithme RRT (Rapidly-exploring Random Tree) est un algorithme d'échantillonnage qui construit un arbre de recherche en explorant de manière aléatoire l'espace des états. Une étude comparative approfondie de ces trois algorithmes en utilisant différents critères d'évaluation tels que la qualité du chemin trouvé, le temps de calcul, la complexité algorithmique, la robustesse face aux changements d'environnement, etc. a été faite et les performances des algorithmes sont mesurées en utilisant des scénarios de navigation complexes et des métriques appropriées.

Mot clés: La robotique, robots mobiles. La planification d'itinéraire, la navigation globale.

Abstract

The objective of our study is to analyse these three algorithms (Dijkstra, RRT, A*) in the context of global navigation for autonomous robots, which involves planning a complete itinerary from a starting point to a destination while avoiding obstacles. The A* algorithm is widely used for its ability to efficiently find the shortest path by using a heuristic to guide the search. The Dijkstra algorithm, on the other hand, finds the shortest path by exploring all the nodes in the graph without using a heuristic. Lastly, the Rapidly- exploring Random Tree (RRT) algorithm is a sampling-based algorithm that constructs a search tree byrandomly exploring the state space. A comprehensive comparative study of these three algorithms using different evaluation criteria such as the quality of the path found, computation time, algorithmic complexity, robustness to environmental changes, etc. has been conducted, and the algorithm performances are measured using complex navigation scenarios and appropriate metrics.

Keywords: Robotics, mobile robots, The route planning, the global navigation.

تلخيص:

الهدف من دراستنا هو تحليل هذه الخوارزميات الثلاث في سياق الملاحة العالمية للروبوتات المستقلة ، والتي تتضمن تخطيط مسار كامل من نقطة على نطاق واسع لقدرتها على العثور بكفاءة على أقصر طريق باستخدام دليل A * البداية إلى الوجهة مع تجنب العقبات. تُستخدم خوارزمية أقصر طريق من خلال استكشاف جميع العقد في الرسم البياني دون استخدام Dijkstra إرشادي لتوجيه البحث. من ناحية أخرى ، تجد خوارزمية هي خوارزمية قائمة على أخذ العينات تقوم (RRT) الكشف عن مجريات الأمور. أخيرًا ، فإن خوارزمية الشجرة العشوائية السريعة الاستكشاف ببناء شجرة بحث عن طريق استكشاف مساحة الحالة بشكل عشوائي. تم إجراء دراسة مقارنة شاملة لهذه الخوارزميات الثلاثة باستخدام معايير تقييم مختلفة مثل جودة المسار الموجود ، ووقت الحساب ، والتعقيد الحسابي ، والمتانة للتغيرات البيئية ، وما إلى ذلك ، ويتم قياس أداء الخوارزمية باستخدام سيناريوهات التنقل المعقدة والمناسبة المقاييس.

الكلمات الرئيسية: الروبوتات ، الروبوتات المتنقلة ، تخطيط الطريق ، الملاحة العالمية.

Introduction générale

Introduction générale :

La robotique est une excroissance de la recherche scientifique impliquant la mécanique, l'électronique et l'informatique. De nos jours, la conception de robots est devenue une réalité dans la société moderne, où cette machine est capable d'effectuer une ou plusieurs tâches de manière autonome. Ces tâches sont généralement, fastidieuses, difficiles, dangereuses, répétitives ou impossibles pour l'homme, tant elles peuvent être simples et mieux réalisées par des personnes.

Le robot est capable de percevoir son environnement en recevant suffisamment d'informations pour lui permettre de déterminer sa position et de trouver indépendamment des chemins efficaces et sûrs pour éviter les collisions et atteindre sa destination. L'asservissement visuel est le processus de contrôle basé sur les informations fournies par une ou plusieurs caméras.

La navigation des robots mobiles peut généralement être définie comme le processus de sélection d'un chemin approprié entre un point de départ et un point d'arrivée et permettant à un robot de le suivre. Les robots mobiles sont devenus une source d'innombrables contributions à la recherche.

La planification consiste à déterminer globalement le meilleur chemin que devrait suivre une entité virtuelle pour se rendre d'un point à un autre dans un environnement complexe (nombreux obstacles, terrain de nature différente). Nous sommes confrontés à ce problème notamment dans le domaine de la robotique mais aussi dans les jeux vidéo. Dans le cadre de ce travail, nous nous sommes intéressés à la navigation de ces agents virtuels au sein de ces environnements virtuels. La planification de trajectoire est une tâche critique pour ces agents car elle affectera directement le réalisme du mouvement et la capacité à opérer dans ces mondes. [1]

Le présent travail consiste à étudier et comparer les algorithmes de navigation globale les plus utilisés afin de les mieux exploiter dans le domaine de la robotique.
Le mémoire est composé de deux parties, la première concerne la partie théorique est formée de deux chapitres : le premier introduit les notions de base de la robotique tandis que le second chapitre traite la navigation globale d'une manière générale et plus particulièrement les algorithmes de recherche de chemin les plus populaires dans la littérature. La deuxième partie traite la contribution de notre travail. La partie st scindée en deux chapitres : le premier concerne la conception et la seconde traite l'implémentation et les résultats de notre système.

Chapitre 01:

La robotique

1. Introduction :

La robotique est un ensemble de technologies qui permettent la conception et la production de machines automatiques ou de robots.

Un robot, selon l'ATILF, peut être vu comme une machine, qui possède des capteurs, et un système de neurones logiques et physiques. C'est matériel. Le bot peut aussi être virtuel [2] (bot automatisé).

La robotique mobile est un domaine dans lequel l'expérience pratique est primordiale. Au début, les robots font leur apparition dans l'industrie grâce aux capacités des manipulateurs : des bras imitant le bras humain et capables d'utiliser différents outils pour accomplir diverses tâches. Au fur et à mesure, ces bras gagnent de nouveaux degrés de liberté à l'aide de plates formes mobiles ; c'est l'apparition des robots mobiles à roues [3].

2. Histoire de la robotique :

L'histoire de la robotique remonte au monde antique. Pendant la révolution industrielle, les humains ont développé la capacité d'ingénierie structurelle de contrôler l'électricité afin que les machines puissent être alimentées par de petits moteurs. Le concept d'une machine semblable à l'homme a été développé au début le vingtième siècle. Les premières utilisations des robots modernes ont eu lieu dans les usines en tant que robots industriels. Ces robots industriels étaient des machines stationnaires capables d'effectuer des tâches de fabrication permettant une production avec moins de travail humain. Des robots industriels programmés numériquement avec intelligence artificielle ont été construits Depuis le XXIe siècle

3. Domaines d'application de la robotique :

L'industrie : Les robots d'usine ont pris en charge la plupart des tâches qui exigent un niveau de précision plus élevé, une vitesse supérieure et de la patience. Les usines automobiles utilisent des robots pour couper et assembler des pièces. Dans les chaines d'assemblage, on retrouve des robots soudeurs, manipulateurs, peintres.

Robot domestique : La robotique domestique met également à la disposition de tout un chacun des appareils d'aide aux ménages, la surveillance.

Robots médicaux : Dans le domaine médical, un robot peut être utilisé pour effectuer des opérations trop délicates pour les mains d'un chirurgien ou comme aide lors d'opérations régulières telles que les pontages coronariens. On parle de surgétique (mot né de l'anglais « surgery » : chirurgie) c'est-à-dire tout ce qui consiste à introduire les derniers outils des technologies informatiques et robotiques dans la pratique médico-chirurgicale. Cette pratique de « chirurgie assistée » est émergente donc bien que peu répandue, elle est en phase de devenir la chirurgie du futur.

Armée automatisée : La technologie robotique est de plus en plus utilisée par les militaires pour les missions de recherche et de sauvetage. La miniaturisation permet aujourd'hui de créer des robots discrets mais dotés de plusieurs capteurs parfaits pour les espionnages et les missions d'infiltration.

Robots scientifiques : Les scientifiques envoient des robots pour explorer les surfaces de la lune ou de planètes comme Mars, tandis que d'autres robots vont dans l'espace pour réparer des équipements spatiaux par exemple pour l'exploration spatiale [4], [5] (aérobot),

RoboCourier [6] : Le système RoboCourier se compose d'un groupe de robots intelligents équipés de capteurs et de logiciels avancés qui leur permettent de naviguer en toute sécurité et efficacement dans les environnements urbains et d'interagir avec les piétons et les obstacles potentiels. Il utilise des robots pour livrer des colis et des expéditions. Ce système vise également à améliorer et accélérer le processus de livraison et fournir un service fiable et efficace aux clients.

Ces robots s'appuient sur des technologies de navigation intelligente, de vision par ordinateur et d'apprentissage automatique pour déterminer les meilleurs itinéraires et éviter les obstacles lors de la livraison.

5. Types de robots :

5.1 Robots industriels : Les robots industriels sont les premiers à avoir été produits en grand nombre. Ils se trouvent plus particulièrement sur les chaînes de montage, et le plus souvent dans l'industrie automobile. Il existe des robotssoudeurs, de démolition, de nettoyage, d'emballage ou de surveillance.

Figure1.1. Robot industriel

5.2 Robots chirurgicaux : Ils sont en fait une autre branche de robots industriels. Ils fonctionnent soit par bras mécaniques reliés à un ordinateur, soit par bras articulés dirigés par le chirurgien et dont les mouvements sont reproduits en même temps.

Figure1.2. Robot chirurgical

5.3 Robots domestiques : Ils peuvent accomplir de multiples tâches ou simplement nous divertir. Plusieurs sociétés ont créé une tondeuse à gazon automatique qui, sur un périmètre donné, peut éviter les obstacles tout en coupant le gazon. Pour nous divertir, une marque très connue, a conçu un robot chien nommé Aïbo. La version la plus récente possède des fonctionnalités très avancées telles qu'appareil photo, reconnaissance vocale …

Figure1.3. Robots domestiques

5.4 Robots militaires : Ils sont principalement utilisés pour la surveillance dans les airs comme dans la mer. Il existe un avion américain sans pilote qui surveille, reconnaît, identifie ou même détruit des cibles ennemies. Il y a également un sous- marin qui a les mêmes propriétés.

Figure1.4. Robot militaire

5.5 Robots explorateurs : Les robots explorateurs remplacent l'homme dans des environnements difficiles. L'exploration de la centrale nucléaire de Tchernobyl a été faite par un robot. L'exploration de l'espace se fait de plus en plus par des robots. En 1976, la sonde Viking 1 s'est posée sur Mars et a pris les premières photos de la planète.

Figure1.5. Robot explorateur

5.6 Robots anthropomorphiques: Les robots anthropomorphiques ont l'apparence humaine, ils sont dotés de la bipédie et sont capables de faire des choses que seul l'humain pouvait faire jusqu'à aujourd'hui. La plus récente version de robot anthropomorphique peut courir à 6 Km/h et peut aussi remplir la tâche de réceptionniste ou de guide d'information. C'est un robot de recherche etil devrait, rendu à terme, être capable de venir en aide à personnes handicapées, âgées ou malades.

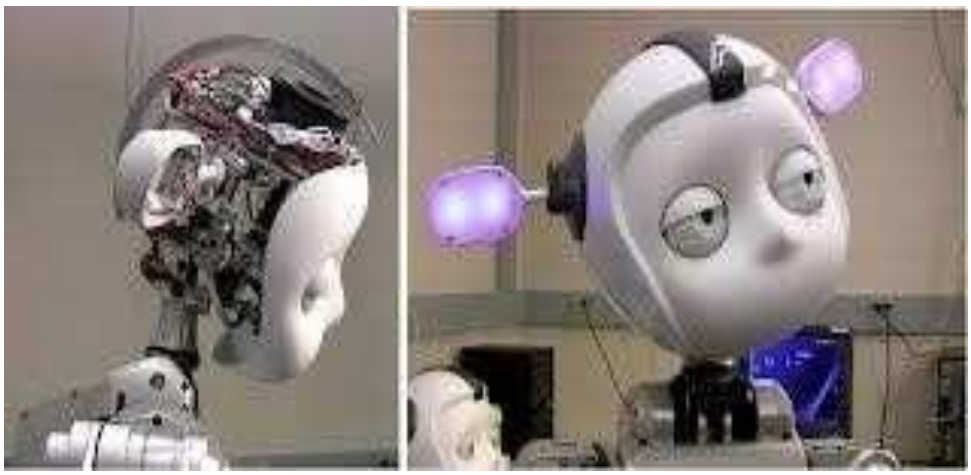


Figure1.6. Robots anthropomorphiques.

6. Objectif des robots :

Le premier but d'un bot est d'effectuer des tâches répétitives et/ou précises. Les robots permettent également d'effectuer des tâches dans des environnements de travail trop dangereux pour l'homme. Les bots peuvent effectuer des tâches automatiques, mais certains d'entre eux ont aussi une certaine intelligence.

7. Travailler dans la Robotique :

Les objets intelligents ont envahi notre quotidien et nous facilitent grandement la vie ! Montres connectées, robots de cuisine, GPS… Tous les secteurs d'activité utilisent aujourd'hui les robots : l'industrie, l'agriculture, mais aussi l'armée et la médecine. Les robots ont pris en charge la plupart des tâches qui exigent un niveau de précision plus élevé, une vitesse supérieure et de la patience.

Derrière ces gadgets et ces machines qui ont révolutionné nos vies se cachent des experts aux connaissances très pointues. Les roboticiens sont généralement des ingénieurs diplômés en électronique, en informatique industrielle ou en automatisme. Zoom sur ce métier d'avenir et sur les **formations** à faire pour devenir un **expert en robotique** avec l'école d'informatique YNOV Campus.

8. Classification:

L'AFRI (Association Française de Robotique Industrielle) distingue quatre classes de robots:

- **Les Télémanipulateurs ou manipulateurs à commande manuelle :** Ils sont commandés à distance et "en temps réel" par un opérateur humain. Cette télécommande se fait à plus ou moins longue distance par signaux mécaniques, hydrauliques, ou le plus souvent électriques. Ces manipulateurs sont employés en forge, fonderie, meulage-ébarbage, milieux "hostiles", etc..., mais nécessitent toujours la présence et l'intervention constante d'un opérateur.
- **Les Manipulateurs automatiques à cycles préréglés :** Leurs mouvements sont limités par des butées et cames réglables à la main. Ils sont commandés à l'aide de logiques à relais ou pneumatiques (séquences fixes), ou par automates programmables et cartes à microprocesseurs (séquences variables). Généralement modulaires, ces appareils sont conçus pour une application déterminée.

- **Les robots programmables :** Ils sont pilotés par des ordinateurs ou des armoires de commande numérique. Leurs mouvements continus dans l'espace sont alors programmés par apprentissage ou en langage symbolique par l'intermédiaire d'un clavier, ou encore sur l'écran d'un poste de CAO. Ils assurent des manipulations complexes, des opérations de soudage, usinage, découpe, peinture et pulvérisation, etc...
- **Les robots dits "intelligents" :** Equipés de capteurs (par exemple un système de vision artificielle ou de suivi de joint en soudage), ils peuvent analyser les modifications de leur environnement ou de leur trajectoire et réagir en conséquence. Ces machines appelées robots de "deuxième génération" commencent à être répandus dans l'industrie. La "troisième génération" disposant de capacités de raisonnement grâce à l'intelligence artificielle fait aujourd'hui l'objet de recherches approfondies.

9. **Composants robotiques de base** :

Les robots sont conçus dans différentes formes et tailles, selon la tâche qu'ils effectueront. Comme transporter des produits, les souder, les peindre ou autre. L'un des types de robots les plus utilisés dans le domaine de l'industrie, et le plus simple en termes de conception, est un robot simple en forme de bras.

Figure1.7 Composants robotiques simples

Le robot se compose des pièces suivantes :

9.1 Bras robotisé : Sa forme est similaire à celle d'un bras humain et il contient des articulations artificielles pour faciliter son mouvement lors de l'exécution des commandes qui lui sont transmises. Selon le but pour lequel le robot est conçu,les bras du robot peuvent varier en taille et en forme, mais ils sont généralement conçus pour imiter le bras humain avec des parties qui ressemblent à l'épaule, au coude et au poignet.

9.2 Répondant final : C'est cette dernière partie du robot qui exécute la tâche émise par le robot, et sa conception dépend de la nature de cette tâche. La pièce répondante peut être une main, un pulvérisateur ou un marteau, et dans les robots médicaux, elle peut être un outil pour suturer les plaies.

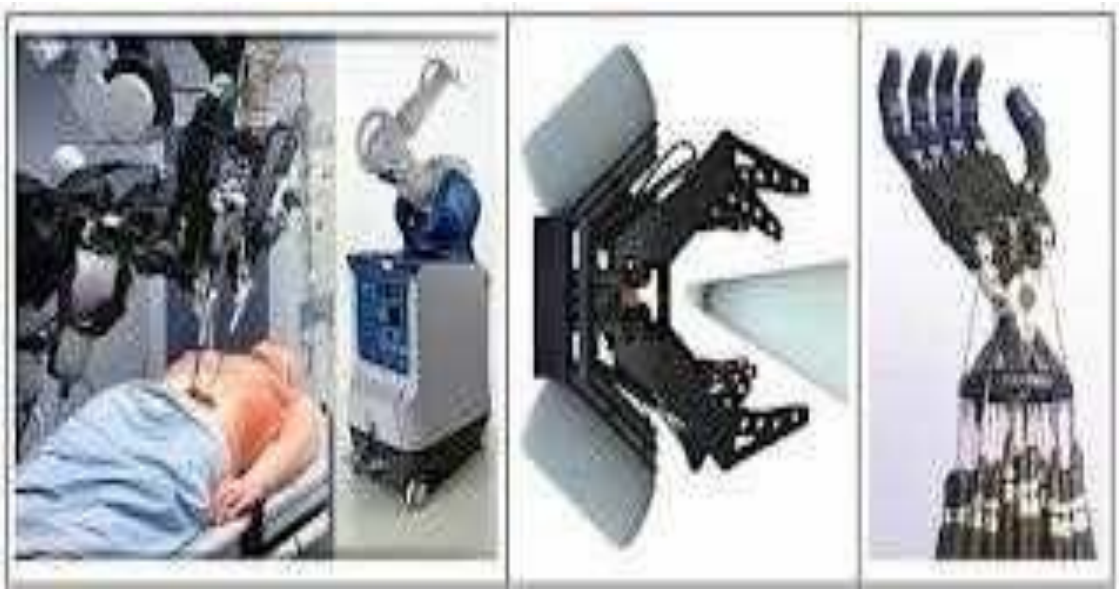


Figure1.8 Une certaine forme d'effecteurs terminaux de robot

9.3 Manette : Le dispositif de contrôle du robot est un ordinateur connecté au robot et c'est le (cerveau) du robot. Il reçoit des données, puis la traite via le code stocké à l'intérieur et donne les commandes nécessaires pour y répondre.

9.4 Machinist : Ce sont les (muscles) du robot, et c'est la partie responsable de son mouvement car il convertit les commandes du contrôleur en mouvement physique. En d'autres termes, les actionneurs simulent l'action des muscles humains pour déplacer des parties du corps d'un robot. Pour être considéré comme un robot, un appareil doit avoir un corps qu'il peut déplacer en réponse à la rétroaction de ses capteurs.

9.5 Capteurs : La fonction des capteurs dans le robot est complètement similaire à la fonction des cinq sens chez l'homme, et c'est le lien entre le robot et l'environnement environnant, car sa fonction est de collecter des données à traiter et à répondre par le robot avec une action précise**.**

Figure1.9. Un groupe de capteurs utilisés dans le robot

Les capteurs les plus populaires utilisés avec les robots sont :

9-5.1 Le capteur tactile, qui détecte le contact entre le robot et tout objet externe, tel qu'un mur, ou entre les parties internes du robot, telles que le bras et la main du robot.

9-5.2 Capteur de distance, Il détecte la distance entre le robot et les objets physiques, En lançant des vagues pour entrer en collision avec le corps et rebondir dessus. En conséquence, il calcule lui-même la distance.

9-5.3 Capteur de lumière, ce capteur détecte l'intensité de la lumière réfléchie par différents objets et distingue leurs couleurs.

9-5.4 Un capteur sonore, comme un microphone, détecte l'intensité des sons environnants et les convertit en impulsions électriques qui sont envoyées au contrôleur du robot.

9-5.5 Source de courant, Pour qu'un robot fonctionne, il doit avoir de la force. Les humains tirent leur énergie de la nourriture. Après avoir mangé, la nourriture est décomposée et convertie en énergie par les cellules du corps humain. La plupart des robots tirent leur énergie de l'électricité. Les bras robotiques fixes peuvent être connectés à une source d'alimentation, comme ceux utilisés dans les usines automobiles, tant qu'ils sont conçus pour fonctionner sur une source d'alimentation externe comme tout autre appareil. Le robot peut également être conçu pour fonctionner sur batteries ou à l'énergie solaire.

10. Conclusion

En conclusion de ce chapitre consacré à la robotique, il est évident que cette discipline connaît une évolution rapide et prometteuse. Les robots jouent un rôle de plus en plus important dans de nombreux domaines, allant de l'industrie à la médecine en passant par l'exploration spatiale. Leur capacité à accomplir des tâches répétitives et dangereuses avec précision et efficacité est indéniable, ce qui ouvre de nouvelles perspectives pour l'humanité.

Nous avons présenté dans ce chapitre les notions de bases sur les robots, ses types, domaines d'applications, etc… dans le prochain chapitre nous discuterons la navigation d'un robot d'une manière générale et les algorithmes les plus utilisés dans la littérature.

Chapitre02

la navigation d'un robot mobile

1. Introduction :

De nos jours, les robots mobiles sont couramment utilisés dans les domaines du divertissement, de la médecine, de l'exploitation minière, du sauvetage, de l'éducation, de l'armée, de l'aérospatiale, de l'agriculture et bien d'autres. Lors de l'exécution de la tâche de navigation, le robot est équipé de divers équipements intelligents nécessaires à la modélisation et au positionnement de l'environnement, au contrôle de mouvement, à la détection d'obstacles et à l'évitement d'obstacles à l'aide de techniques de navigation.

La position initiale par rapport à la position cible est la fonction la plus importante de toute technologie de navigation. Par conséquent, la sélection correcte de la technologie de navigation est l'étape la plus importante dans la planification de la trajectoire du robot lorsqu'il opère dans des environnements simples et complexes. De nombreuses technologies ont été développées par divers chercheurs dans le domaine de la navigation robotique mobile et c'est le sujet le plus étudié aujourd'hui. Navigation est classée en trois catégories : Navigation globale Navigation locale et navigation personnelle.

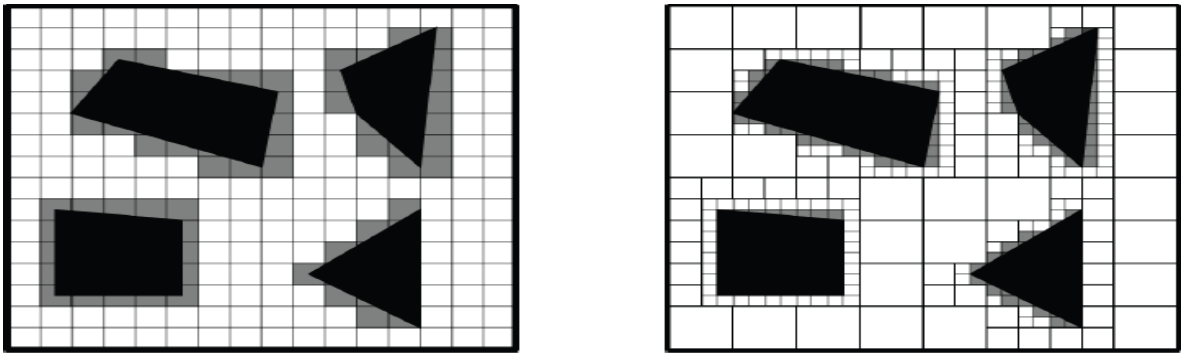
La possibilité de positionner des éléments dans l'environnement par rapport à un axe de référence et de se déplacer vers une cible prédéterminée est la navigation globale, dont je vais parler. La navigation locale consiste à déterminer les conditions dynamiques de l'environnement et à établir des relations de position entre divers éléments. Traiter les divers éléments de l'environnement les uns par rapport aux autres, en considérant leur emplacement, est une navigation personnelle.

2. Représentations topologiques de l'environnement :

2. 1 Représentations approximatives de l'environnement :

Modèles à base de grilles :

- Le premier modèle de représentation approximative utilise des grilles régulières
- L'environnement est pavé par des cellules, qui peuvent avoir trois états :



(a) Grille régulière (b) *Quadtree* de profondeur 3

Figure2.1. Exemple sur la grille.

- La précision de la représentation obtenue dépend directement de **la taille des cellules** utilisées.
  - **L'occupation mémoire** de cette méthode constitue ainsi son premier point faible.
- Une évolution de ce modèle est apparue sous la forme des **grilles hiérarchiques**.
  - Décrire l'espace navigable par une succession de grilles de plus en plus précises, organisées sous forme d'**arbre**.
- Fortement utilisée en animation comportementale du fait de leur **simplicité** de mise en œuvre et de leur rapidité d'exploitation.

- Les grilles sont très difficilement généralisables à des **environnements quelconques** (des obstacles non alignés sur les axes, un cadre de simulation où l'individu devra se déplacer finement).

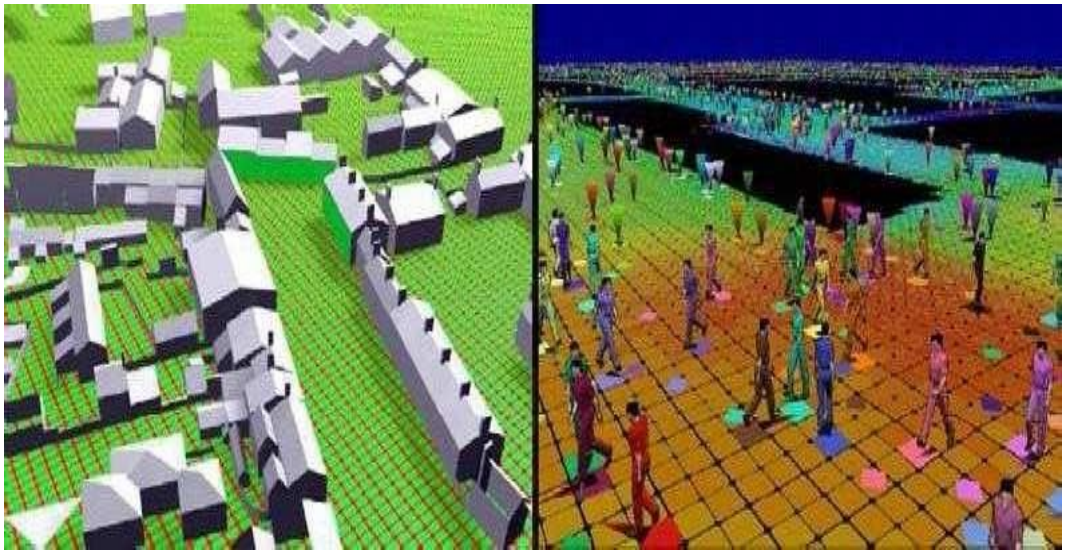
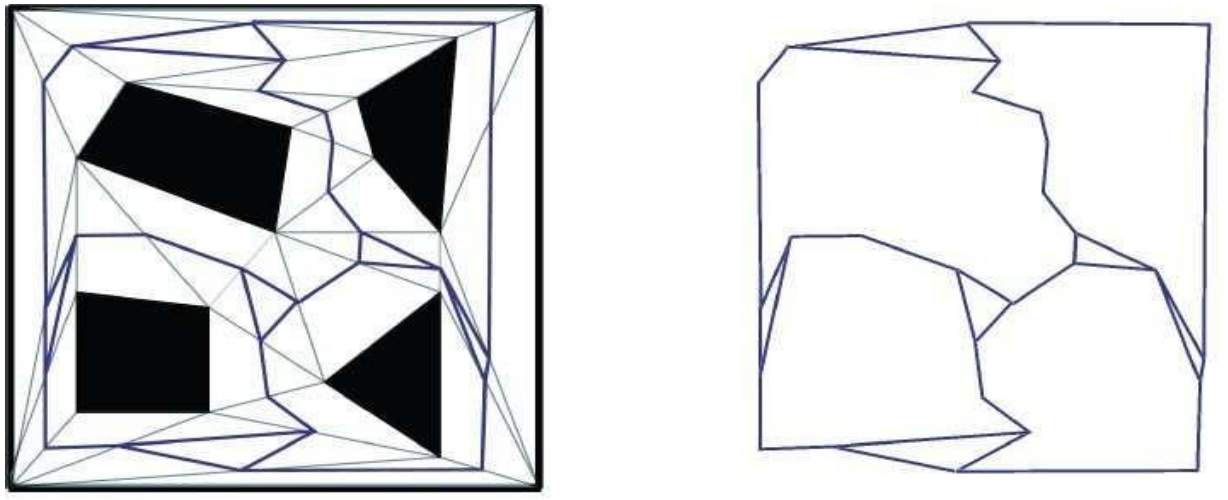


Figure2.2. Les grille sur l'environnement.

Cartes de cheminement :

- Discrétiser l'espace navigable sous la forme d'un **réseau de chemins**.
  - Ce réseau est obtenu en reliant des points clefs répartis à l'intérieur de l'environnement.
- Plusieurs méthodes existent pour créer des cartes de cheminement, différentes dans la manière de créer et de relier ces points clefs.



(a) Calculs sur l'environnement d'origine (b) Exemple de carte de cheminement

Figure2.3. Carte de cheminement.

- Fournir une description **très condensée** de l'environnement, ne nécessitant une prise de décision qu'au niveau des points clefs.
- Leur exploitation devient très difficile lorsqu'il s'agit de gérer le **croisement des entités** le long d'un même chemin.
- Leur fonctionnement est **trop éloigné** du comportement humain pour être exploité dans des simulations réalistes.

Représentations exactes de l'environnement :

- Organiser les données spatiales tout en **conservant intégralement** les informations qu'elles contiennent à l'origine.
- Découper l'espace navigable en **cellules convexes** de différentes formes.
  - Il existe **plusieurs modèles** pour effectuer cette subdivision.

- **Triangulation de Delaunay**
  - Créer un ensemble de triangles en réunissant des points fournis en entrée.

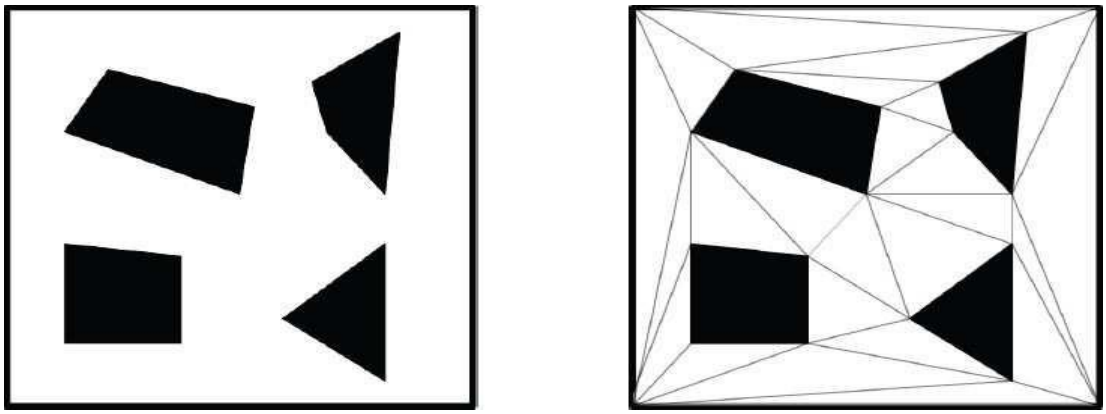


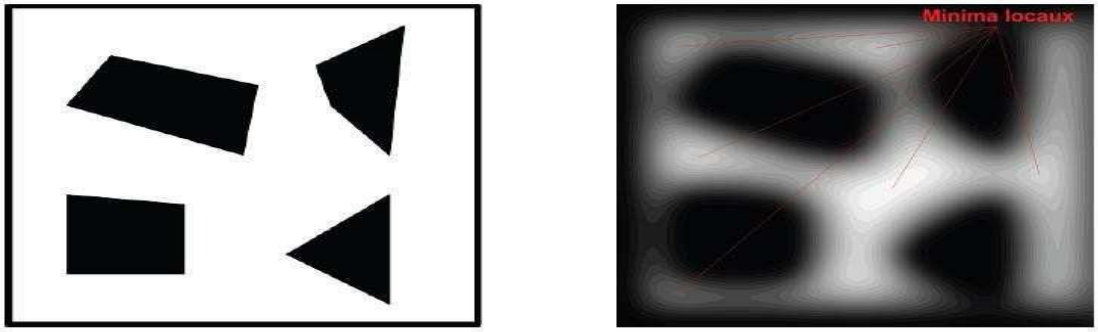
Figure 2.4 : Triangulation de Delaunay contrainte

Ces représentations reproduisent **exactement** l'environnement d'origine tout en l'organisant de manière plus accessible.

- Pour des environnements à géométrie complexe, notamment contenant des courbes, le **nombre de triangles** obtenus **augmente** très rapidement.

Le modèle de champs de potentiels :

- Les champs de potentiels caractérisent les obstacles de l'environnement par des forces de répulsion, et les buts par des forces d'attraction.



(a) Environnement d'origine (b) Exemple de champs de potentiels

Figure2.5 Champs de potentiels.

- Un gradient de forces, ou champ de potentiel, est déduit en chaque point de l'environnement comme étant une somme pondérée, le plus souvent par la distance, des potentiels de répulsion et du potentiel lié au but.
- La navigation d'une entité est alors simulée par une descente de gradient depuis sa position dans l'environnement.
- Les méthodes basées sur les champs de potentiels s'avèrent **simples et efficaces**.
- Le problème des minima locaux : zones de l'environnement où un potentiel minimal isolé apparaît.
    - Pousser l'entité à se déplacer vers le minimum local le plus proche, qui ne représente pas forcément son but.

3. La navigation d'un robot mobile :

Est une tâche qui consiste à naviguer généralement libre dans l'espace de configuration (environnement de travail) sans collision avec des obstacles proches du robot.

Intérieure :

Le moyen le plus simple d'amener un bot à un emplacement objectif est de simplement le pointer vers cet emplacement. Ce routage peut être mis en œuvre de différentes manières : enterrer une boucle inductive ou un aimant dans le sol, tracer des lignes au sol, placer des balises, des panneaux, des codes-barres, etc. Dans l'environnement. Ces véhicules à guidage automatique (AGV) sont utilisés dans des scénarios industriels pour des tâches de transport. Le robot peut être piloté à l'aide de dispositifs de positionnement internes.

Il existe une grande variété de systèmes de navigation intérieure :

Navigateur AVM, AVM Navigator est un module plug-in RoboRealm qui permet la reconnaissance d'objets et la navigation autonome à l'aide d'une seule caméra vidéo sur un robot

Le navigateur crée un itinéraire entre l'emplacement actuel et la position cible sous la forme d'une série de coordonnées. Si la direction actuelle du robot n'indique pas le prochain waypoint, le navigateur fait pivoter le corps du robot.

Lorsque le robot atteint un waypoint, le navigateur change de direction et se déplace vers le waypoint suivant dans la chaîne, et ainsi de suite jusqu'à ce que la position cible soit atteinte.

Navigation externe :

Certains algorithmes de navigation extérieure modernes sont basés sur des réseaux neuronaux convolutifs et l'apprentissage automatique, qui sont capables d'une inférence précise étape par étape.

Contrôleurs de vol autonomes :

Les contrôleurs de vol autonomes open source typique ont la capacité de voler en mode entièrement automatique et d'effectuer les opérations suivantes :

- Décollez du sol et volez à une altitude spécifiée.
- Déplacez-vous vers un ou plusieurs waypoint.
- Rotation autour d'un point précis.
- Retournez au site de lancement.
- Descendez à une vitesse spécifiée et atterrissez.
- Le contrôleur de vol embarqué s'appuie sur le GPS pour la navigation et le vol stable, et utilise souvent des systèmes d'augmentation par satellite supplémentaires (SBAS) et un capteur altimètre (pression barométrique).

4. Problème de navigation :

La mobilité dans les robots mobiles est traditionnellement comprise comme une solution au problème proposé par ces questions suivantes (Levitt [7]) :

- Où suis-je?
- Où sont les autres endroits liés à moi ?
- Comment puis-je me rendre à d'autres endroits à partir d'ici ?
- Comment est la structure de l'environnement dans lequel je me trouve ?

Répondre à ces questions faites référence à la traduction pour où se trouve le robot, à la planification d'itinéraire pour voir comment se rendre à d'autres endroits

d'où se trouve le robot et à la navigation pour effectuer l'itinéraire commandé par le système de planification d'itinéraire.

Cette approche a dominé le paradigme de la navigation robotique et plusieurs solutions réussies ont été proposées. Cependant, les avancées dans les développements techniques permettent une réflexion plus large sur la navigation robotique mobile en apportant une solution à un problème "plus important".

La réponse à ces nouvelles questions porte sur l'importance de la perception pour déterminer l'emplacement et les éléments d'un lieu dans lequel se trouvent un robot et l'importance de la modélisation environnementale pour déterminer la structure de l'espace et déduire les connexions entre les lieux. Comme les robots sont intégrés dans des environnements humains et vivent avec des humains, la perception et la modélisation gagnent en importance pour permettre les futures tâches liées aux robots de service telles que la manipulation ou l'interaction homme-robot.

Un autre aspect important est la collecte d'informations sur l'environnement, qui doivent être disponibles pour les systèmes de navigation, entre autres tâches que les robots mobiles effectuent. Cette information est fournie par un système perceptif, et ainsi les problèmes non triviaux de détection d'objet et de reconnaissance de lieu se posent. L'apparence du lieu est l'un des problèmes les plus difficiles avec la reconnaissance de scène. Parfois, le même endroit peut sembler différent ou des endroits différents peuvent se ressembler. De plus, la position du robot et la variété de son point de vue peuvent affecter le placement lors de la revisite. Les modèles d'environnement et la façon dont les tâches de navigation sont définies peuvent être reconfigurés en tenant compte des nouvelles technologies et tendances. Cela donnera plus d'autonomie aux robots mobiles et les aidera à interagir avec les humains dans leurs environnements habituels. Dans ce sens, la vision est le principal capteur de navigation, de localisation et de reconnaissance de scène. La reconnaissance de lieu est un problème difficile bien connu, non seulement avec la façon dont un robot donne le même sens qu'un humain donnerait à la même image, mais aussi avec le contraste dans la façon dont ces images apparaissent dans le monde réel. La reconnaissance de lieu a une relation étroite avec plusieurs domaines majeurs de la recherche en robotique.

5. Techniques de navigation utilisées pour la navigation des robots mobiles :

Depuis plusieurs décennies, divers chercheurs et scientifiques ont fourni de nombreuses méthodologies sur les approches de navigation. Diverses méthodes employées pour la navigation d'un robot mobile sont globalement classées en deux catégories, à savoir les approches classiques et réactives.

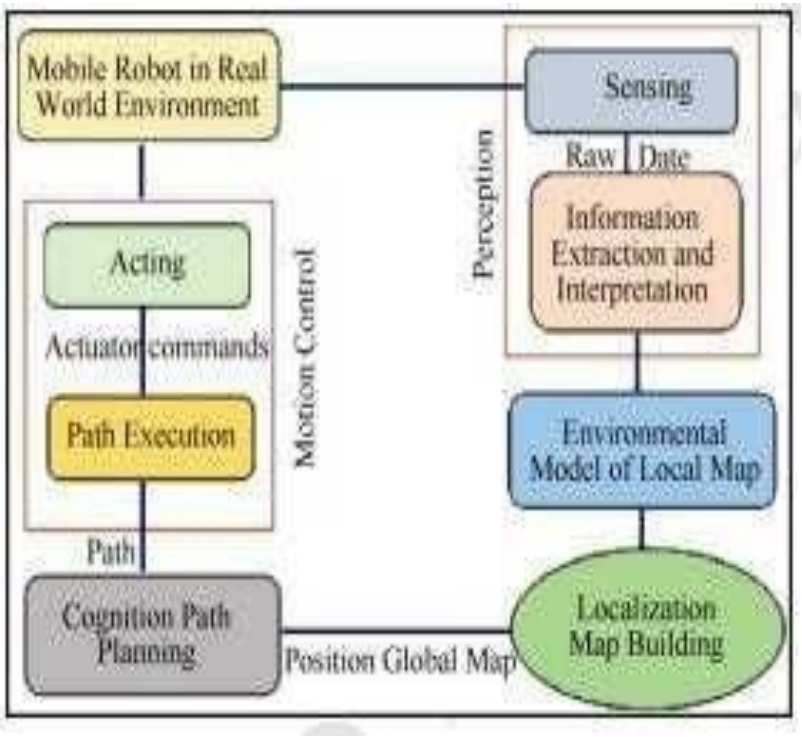


Figure2.6. Diagramme de flux pour la navigation d'un robot mobile[8]

Approches classiques :

Dans le domaine de l'intelligence artificielle, les approches classiques étaient couramment utilisées pour résoudre les problèmes de contrôle de mouvement des robots avant le développement de nouvelles technologies. Cette approche a l'avantage d'obtenir des résultats clairs, soit d'obtenir le résultat souhaité, soit de confirmer qu'il n'y a pas de résultat, mais elle présente des inconvénients tels qu'un coût de calcul élevé et un manque de réponse à l'incertitude de l'environnement, et par conséquent elle n'est pas préférée pour de vrais -temps d'utilisation. Certaines approches classiques telles que CD, RA et APF sont passées en revue.

Approche de décomposition cellulaire (CD) :

L'approche de décomposition de cellules (CD) divise une région en grilles (cellules) non superposées et utilise des graphes de connectivité pour se déplacer d'une cellule à une autre afin d'atteindre un objectif. Les cellules contenant des obstacles sont divisées en deux nouvelles cellules pour obtenir une cellule pure, qui est ajoutée à la séquence pour déterminer le chemin optimal de la position initiale à la position cible.

L'approche de décomposition de cellules peut être adaptative, approximative ou exacte. Dans l'approche de décomposition de cellules exacte, les cellules n'ont pas de forme ou de taille spécifique, mais peuvent être déterminées par la carte de l'environnement et la forme des obstacles. Les éléments de l'environnement sont décomposés en petits éléments (trapézoïdaux et triangulaires), puis numérotés. Chaque élément dans l'environnement agit comme un nœud pour un graphe de connectivité. Les nœuds adjacents sont autorisés à rejoindre l'espace de configuration pour former un chemin qui relie les positions de départ et d'arrivée. Ce chemin est ensuite converti en un chemin libre en reliant les points médians des intersections des cellules adjacentes.

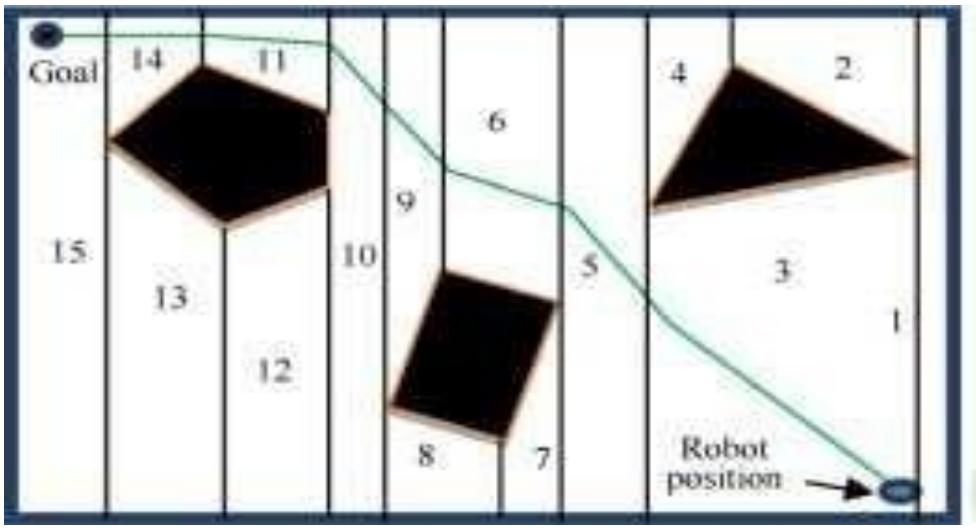


Figure2.7 Décomposition exacte des cellules.

Approche de feuille de route (AR):

RA est également connu sous le nom d'approche de l'autoroute. C'est la façon d'aller d'un endroit à un autre et de se connecter entre les espaces libres représentés par un ensemble de courbes unidimensionnelles [8], les nœuds jouent un rôle important dans l'obtention de la trajectoire souhaitée pour le robot. RA est utilisé pour trouver le chemin le plus court entre la position initiale du robot et sa position cible ; La figure 2.8 représente l'histogramme de visibilité où la zone assombrie montre les obstructions et la ligne pointillée montre la trajectoire respective de la position initiale à la position finale [9].

Cette méthode est également utilisée dans les environnements avec des obstructions polygonales ce qui représente les sommets du polygone avec les nœuds et les arêtes comme lien entre les nœuds [10]. Le diagramme de Voronoi [11–13] est un autre algorithme de feuille de route utilisé pour la planification d'itinéraire de robot, Cette méthode divise la région en sous-régions où tous les bords de la forme sont créés à l'aide de points équidistants des deux points adjacents sur la limite de l'obstacle. La figure 2.9 représente le fonctionnement d'un diagramme de Voronoi.

L'approche hybride a été développée en combinant le graphe de vision, le diagramme de Voronoi et la méthode du champ de potentiel [14] pour obtenir la trajectoire optimale. Il a été noté que l'approche ne parvient pas à obtenir le chemin optimal et le processus de mise en œuvre est complexe. Pour améliorer le processus de recherche du chemin le plus court, Sanchez et al ont légèrement modifié l'approche Probabilistic Roadmap (PRM) [15]. Dans leur approche, une stratégie de validation de collision lente avec PRM a été introduite pour résoudre le problème de planification de trajectoire dans l'environnement réel. Naviguer dans l'environnement 3D d'un véhicule aérien sans pilote Il a été testé avec succès par Yan et al. [16]. Dans cette approche, l'approche feuille de route est présentée avec probabilité rédaction pour contrôler la trajectoire de vol.

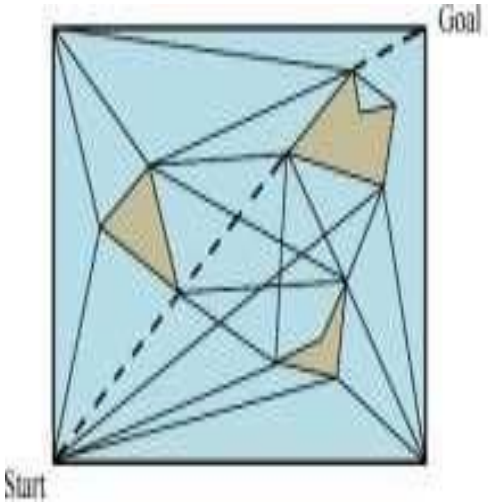


Figure2.8. Graphique de visibilité.

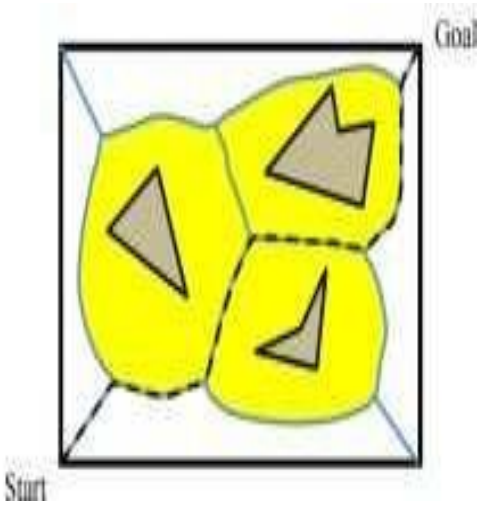


Figure2.9 Diagramme de Voronoi.

Approche du champ de potentiel artificiel (APF):

Al-Khatib en 1986 a introduit l'approche APF de la navigation robotique mobile [17]. Selon lui, la cible et les obstacles agissent comme des surfaces chargées et le potentiel total crée la force imaginaire sur le robot. Cette force imaginaire attire le robot vers la cible et l'éloigne de tout obstacle comme la montre la figure2.8. Ici, le robot suit la pente négative pour éviter l'obstacle et atteindre le point cible. Certaines improvisations sont faites dans la méthode APF en utilisant des lois électrostatiques [18]. La mise en œuvre de l'électrostatique permet de produire la fonction de potentiel et Déterminez le chemin sans collision en temps réel.

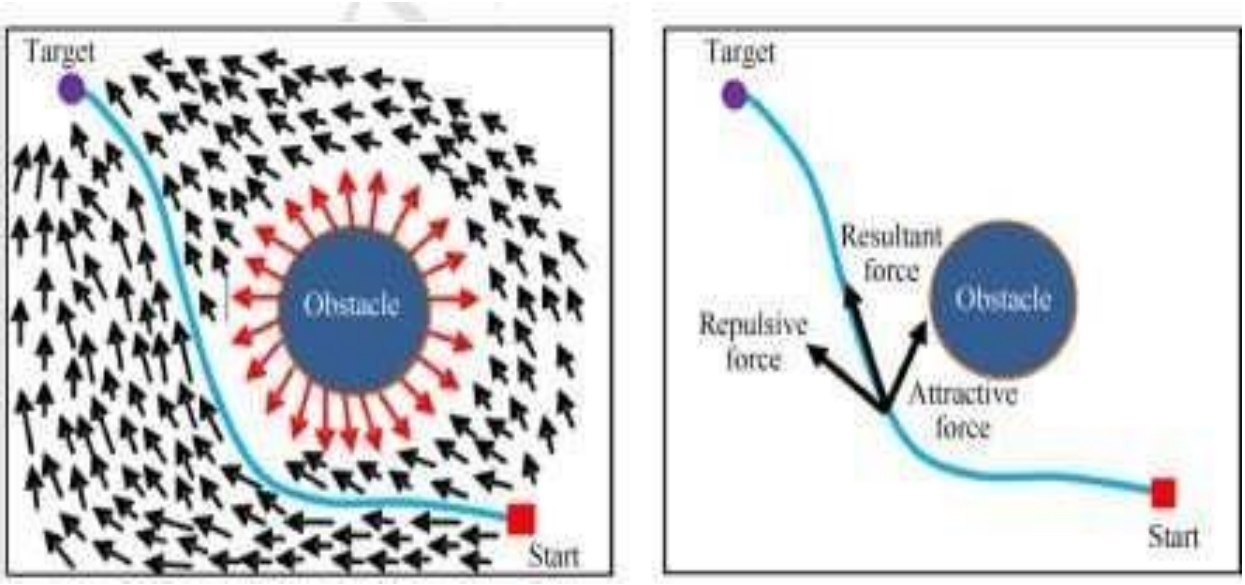


Figure2.10 Navigation robot mobile par approche APF.

Approche réactive :

En 1986, Brooks a proposé une approche réactive qui se distingue par l'abandon des phases de modélisation et de planification [19]. Le contrôle du robot se fait alors sans utiliser le modèle de l'environnement (figure2.10). Les stratégies de navigation réactives n'utilisent que les valeurs courantes des capteurs et non des données provenant d'un modèle interne, pour décider de l'action à effectuer. Le principe de l'approche réactive est basé sur la décomposition de la tâche de navigation en un ensemble de comportements actifs de base (explorer, aller au but, éviter les obstacles, suivi des murs,…).

Pour guider le robot, il faut donc choisir à chaque instant lequel de ces comportements est à activer? Ce problème est connu dans la littérature scientifique sous le nom de sélection de l'action. La solution proposée par Brooks est de diviser la tâche globale en un ensemble des sous tâches secondaires. Cette architecture est appelée "subsumption", utilise une hiérarchie des comportements qui se déclenchent donc selon un ordre de priorité en fonction des données de perception (fusion des comportements).

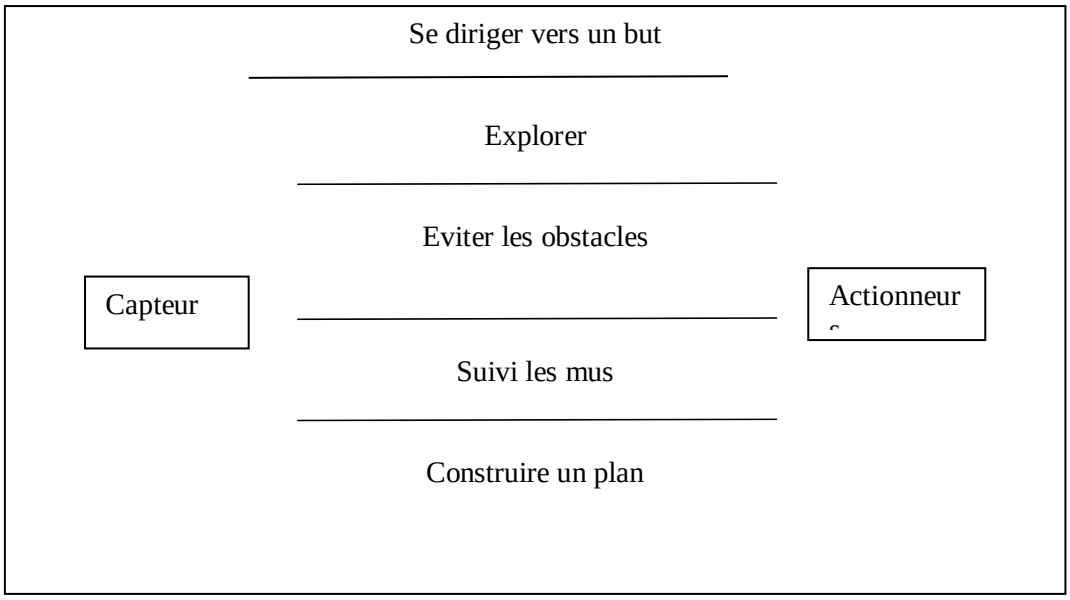


Figure2.11 Architecture réactive.

La figure 2.11 décrit les différentes couches comportementales d'un robot en utilisant la structure subsumption. Cette architecture peut être définie comme une hiérarchie de comportements plutôt que comme une hiérarchie fondée sur une abstraction des données. Le concept de base est le suivant : si complexe soit-il, peut être décomposé en comportements élémentaires représentant chacun un niveau de compétence. Ces différents niveaux sont placés en couches parallèles et chaque compétence couple directement la perception avec l'action. Une telle architecture a, au moins, trois avantages :

D'abord, elle peut réagir aux éventualités en temps réel dû au parallélisme.

Elle présente une bonne robustesse ; le système peut fonctionner même si un ou plusieurs comportements échouent. Elle apporte donc l'avantage d'avoir une trajectoire qui s'adapte aux changements de l'environnement.

Cette architecture est une approche purement réactive vient de son manque de capacités de raisonnement de haut niveau et de modularité [20, 21].

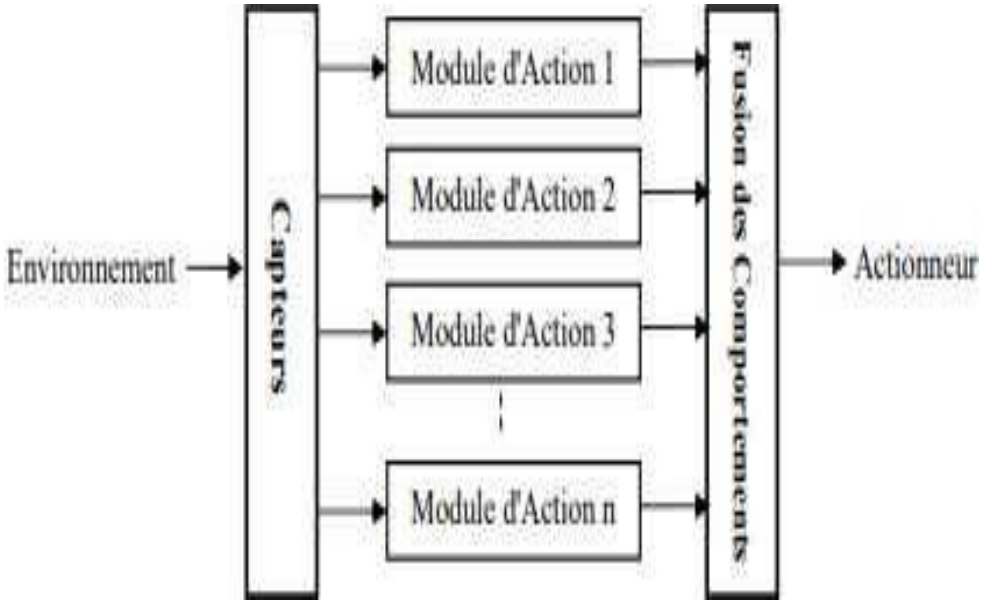


Figure2.12 Architecture modulaire de Subsumption.

En effet, sans représentation interne de l'état de l'environnement, il est très difficile de planifier une suite d'actions en fonction d'un but à atteindre. Les robots utilisant cette architecture sont donc en général efficaces pour la tâche précise pour laquelle ils ont été conçus et dans l'environnement pour lequel ils ont été prévus, mais sont souvent difficiles à s'adapter à une tâche différente. Puisque, le robot n'a qu'une vue très réduite de son environnement, ces architectures ne peuvent garantir le succès de la mission en toute circonstance. Les réussites de ces architectures sont liées au couplage direct entre la perception et l'action qui permet une prise en compte très rapide des phénomènes dynamiques de l'environnement et donc une robustesse et fiabilité dans les situations complexes.

6. Approches de planification globale :

C'est la modélisation de l'espace de l'environnement par un graphe, ou la recherche de la trajectoire est basée sur l'utilisation des algorithmes des graphes, on peut citer: le graphe de visibilité, la décomposition cellulaire…etc. [22, 23].

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.

Il existe un groupe d'algorithmes, nous mentionnons les plus courants d'entre eux :

L'algorithme de Dijkstra :

L'algorithme de Dijkstra est un algorithme qui résout le problème de trouver le chemin le plus court entre deux nœuds dans une déclaration qui ne contient pas de liens pondérés négativement. L'algorithme est utile dans de nombreuses applications, telles que la recherche de l'itinéraire le plus court entre deux villes surune carte, où les poids de jonction peuvent représenter la longueur de la rue, le niveau de congestion sur cette rue, leur somme ou tout autre critère approprié. L'auteur de cet algorithme est le hollandais Edscher Dijkstra en 1959.

Les usages:

Cet algorithme est principalement utilisé dans les applications de calcul d'itinéraire telles que Google Maps. Grâce à cet algorithme, le chemin le plus court entre deux emplacements est trouvé. Où le réseau routier est représenté sous la forme d'un graphique et les sites sont sous la forme de nœuds ou de points dans ce réseau, puis la distance la plus proche dans ce réseau entre deux points quelconques est calculée.

Permet de trouver l'ensemble des meilleurs chemins entre deux nœuds dugraphe.

- L'intérêt majeur de cet algorithme est sa robustesse.
- Son point faible est sa complexité calculatoire.

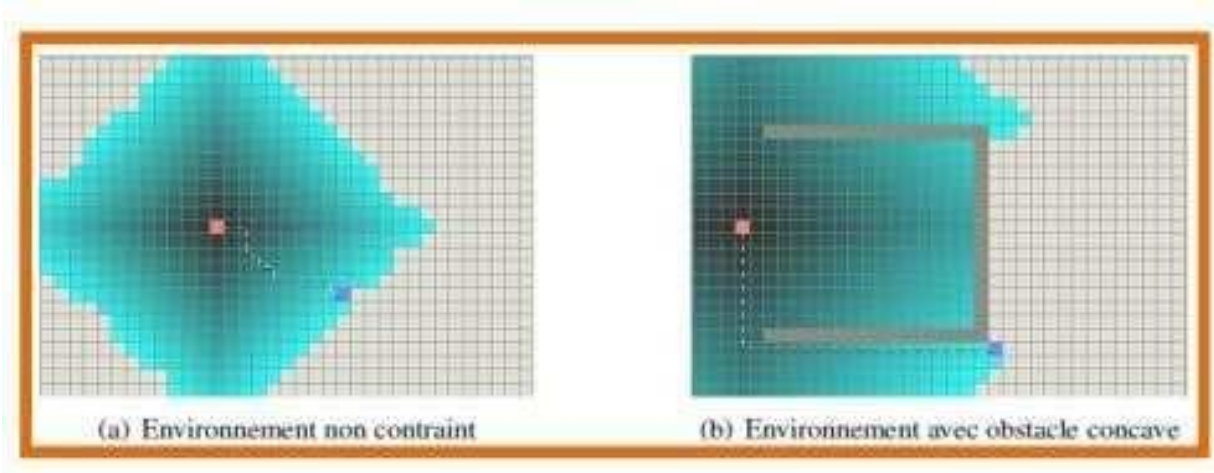


Figure2.13 : Algorithme Dijkstra.

L'algorithme A* :

L'algorithme de recherche A* est une simplification de l'algorithme de Dijkstra introduit par Peter Hart en 1968[24][25][26] .

Il s'agit d'un algorithme sophistiqué de l'algorithme Breadth-First Search, qui est parfait.

Par idéalisme, cela signifie qu'il sortira avec le chemin le moins cher (donne la solution optimale) et le perfectionnisme, c'est-à-dire qu'il trouvera tous les chemins disponibles qui font le lien entre le point de départ et l'écurie (toutes les solutions qui remplissent une condition).

Entrées et sorties d'algorithme : Il n'y a qu'une seule formule qui suffit, et c'est le cœur et l'âme de cet algorithme : **F = G + H.**

Quelles sont ces trois variables et pourquoi sont-elles si importantes ? Continuons à trouver la réponse.

G : est le coût de déplacement d'un nœud à un autre, cette variable varie d'un nœud à l'autre.

H : est la fonction estimée (indicative) entre le nœud courant et le nœud cible. Cette valeur n'est pas réelle, mais elle est plus proche de la vérité calculée entre le point de départ et le stable (source et destination).

F : est la somme des deux variables précédentes et exprime le moindre coût pour passer d'un nœud à la base suivante de manière optimale.

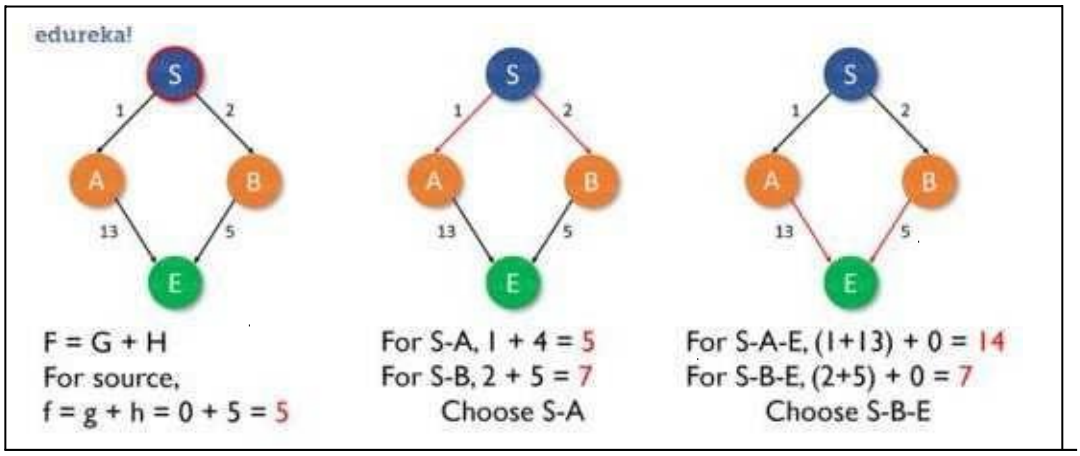


Figure2.14 exemple de A*

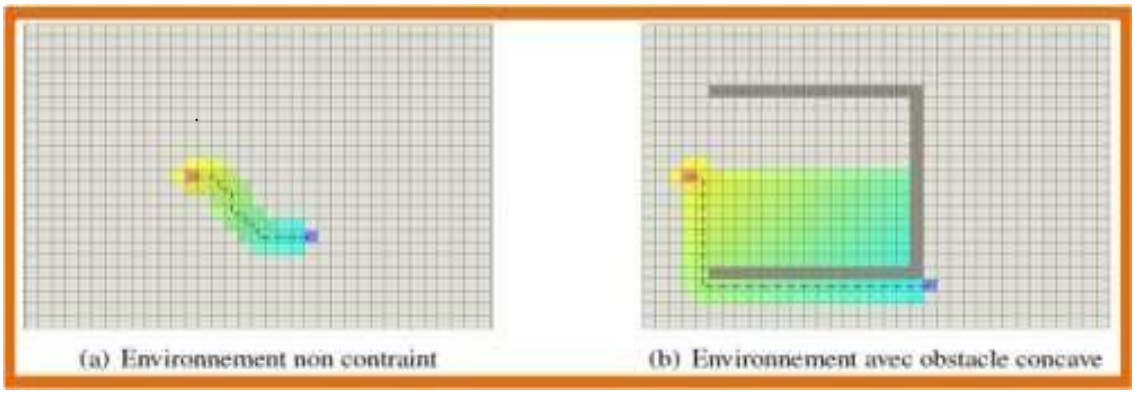


Figure2.15: ALGORITHME A*

- fonctionne d'une façon similaire à l'algorithme de Dijkstra, mais en ajoutant un coût prédictif aux nœuds correspondant au reste du chemin à parcourir.

- L'avantage de cet algorithme réside dans sa rapidité calculatoire.
- Son inconvénient majeur réside dans le recours à une heuristique.

Algorithm RRT (Les Rapidly-exploring Random Trees):

Est une modification de l'algorithme de recherche traditionnel, l'idée principale est d'améliorer et d'utiliser les performances de l'algorithme traditionnel (Arbre découverte aléatoire) pour résoudre les problèmes de planification de chemin.

Ceci a été réalisé en prenant en considération la direction de recherche de l'algorithme de base et en l'orientant vers des coordonnées spécifiques.

Les résultats de laboratoire du système de simulation de ce développement ont montré des résultats prometteurs et des orientations pour les recherches futures.

Méthode:

Algorithme de planification de chemin basé sur RRT (Rapidly Explore RandomTree): En détectant la collision de points d'échantillonnage dans l'espace d'état, la modélisation spatiale est évitée et l'espace de grande dimension et les chemins contraints complexes peuvent être résolus efficacement. Problèmes de planification. L'avantage de cette méthode est que l'espace de grande dimension peut être recherché rapidement et efficacement. Grâce aux points d'échantillonnage aléatoires dans l'espace d'état, la recherche est dirigée vers la région vide, de manière à trouver un chemin planifié du point de départ au point cible.

Planification de parcours dans des environnements environnementaux et dynamiques. Semblable à PRM, cette méthode est complète et sous-optimale.

RRT est une méthode de cartographie efficace dans l'espace multidimensionnel. Il utilise un point initial comme nœud racine et ajoute des nœuds feuilles par échantillonnage aléatoire pour générer un arbre couvrant aléatoire. Lorsque les nœuds feuilles de l'arbre aléatoire contiennent le point cible ou entrent dans la zone cible, un arbre aléatoire peut être trouvé dans l'arbre au hasard. Le chemin du point initial au point cible.

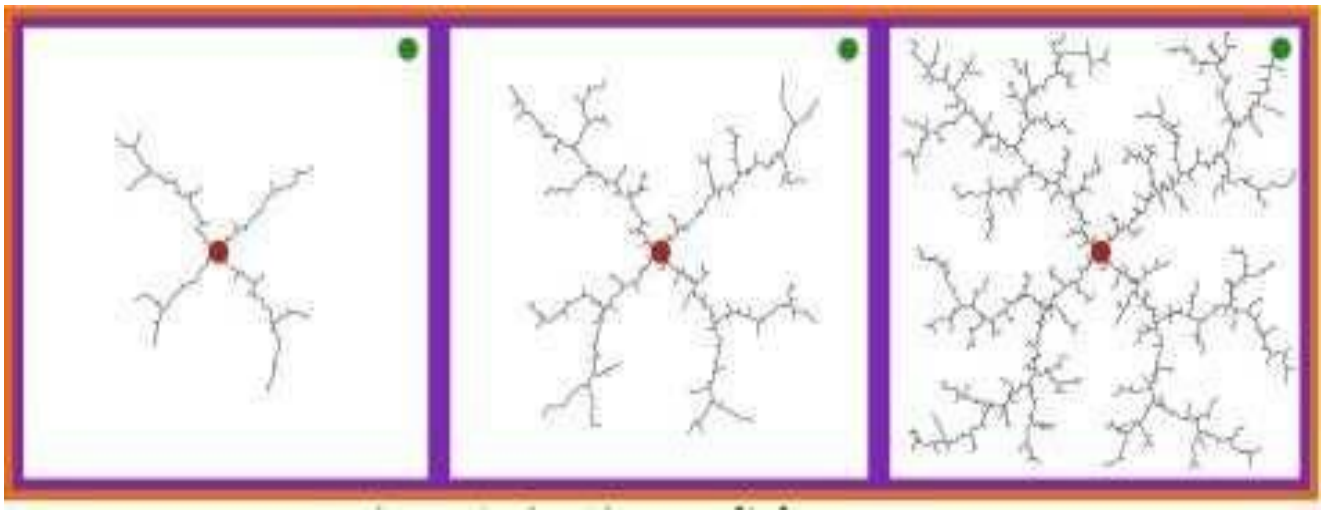


Figure 2.16 algorithme RRT

7. Comparaison des algorithmes :

La comparaison des algorithmes est décrite dans le tableau ci-dessous :

Algorithme	complexité	Structure données	heuristique	Temps obstétrical
DIJKSTRA	N^2	Listes	Pas d'heuristique	Le temps de génération est plus longque A*
A*	Nlog(n)	File de priorité	Manhattan/euclidienne	Longueur globale optimisée et temps de génération plus court.
RRT	Nlog(n)	Arbre	Pas d'heuristique	temps de générationcourt.

Tableau 2.1 : Comparaison des algorithmes.

8. Conclusion :

Dans ce chapitre, nous avons traité de la méthode de planification de trajectoire pour la navigation du robot, en commençant par la façon de représenter l'environnement, car il existe plusieurs façons de représenter l'environnement.

A partir de là, nous avons choisi trois algorithmes parmi les algorithmes généraux de navigation pour robots, nous avons présenté une définition pour chacun d'eux et une tentative de trouver le principe de chacun d'eux pour savoir quels algorithmes sont adaptés afin de planifier un itinéraire à moindre coût. Nous avons également présenté un tableau montrant certaines des différences.

Chapitre03

Conception du système

1. Introduction :

L'étude vise à analyser ces trois algorithmes dans le contexte de la navigation globale des robots autonomes, qui consiste à planifier un itinéraire complet d'un point de départ à un point d'arrivée tout en évitant les obstacles. L'algorithme A* est largement utilisé pour sa capacité à trouver efficacement le chemin le plus court dans un graphe pondéré, en utilisant une heuristique pour guider la recherche. L'algorithme de Dijkstra, quant à lui, trouve le chemin le plus court en explorant tous les nœuds du graphe, sans utiliser d'heuristique. Enfin, l'algorithme RRT (Rapidly-exploring Random Tree) est un algorithme d'échantillonnage qui construit un arbre de recherche en explorant de manière aléatoire l'espace des états.

L'objectif principal de ce chapitre est la mise en œuvre d'une étude comparative approfondie de ces trois algorithmes en utilisant différents critères d'évaluation tels que la qualité du chemin trouvé, le temps de calcul, la complexité algorithmique, etc. Les performances des algorithmes sont mesurées en utilisant des scénarios de navigation complexes et des métriques appropriées.

La phase de conception est réalisée à travers un processus itératif de définition des différents composants et modules du système et de leurs interactions. Une bonne conception est essentielle pour développer un logiciel efficace. Un système bien conçu est facile à construire, facile à entretenir et facile à comprendre.

L'étude porte sur les méthodes RRT, A* et Dijkstra appliquées à la recherche de chemin à l'aide d'un robot. Pour le diagramme de chemin probabiliste, il est supposé compiler un groupe des composants principaux de ce groupe, qui sont la détection de collision et la recherche du plus proche voisin.

2. Modèle de la navigation :

2-1. Conception générale de notre système :

Dans cette section, nous visons à présenter la structure générale de notre système proposé, cette structure est illustrée dans le diagramme schématique de la Fig3.1. La première étape consiste à représenter l'environnement, dans la deuxième étape consiste à effectuer une mise en page à l'aide de RRT, A * et Dijkstra.

Le schéma suivant est résumé dans la réalisation de la planification du trajet de l'entité virtuelle.

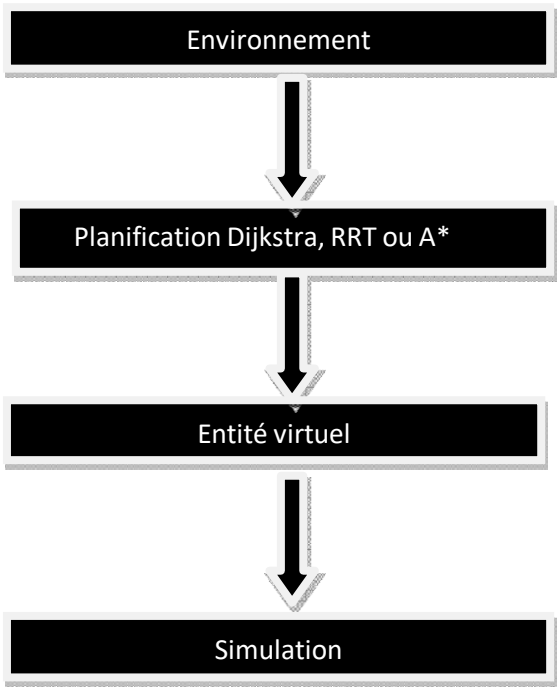


Figure3.1 Schéma générale de l'application

2-2. Conception détaillée de notre système :

2-2-1. Représentation l'environnement et planification de chemin :

La représentation de l'environnement et la planification de chemin dans notre système pour but de le raffiner et l'enrichir en précisant exactement d'où l'entité virtuel va passer dans chaque sous-espace de l'environnement.

2-2-2. Schéma d'algorithmes A*, Dijkstra et RRT :

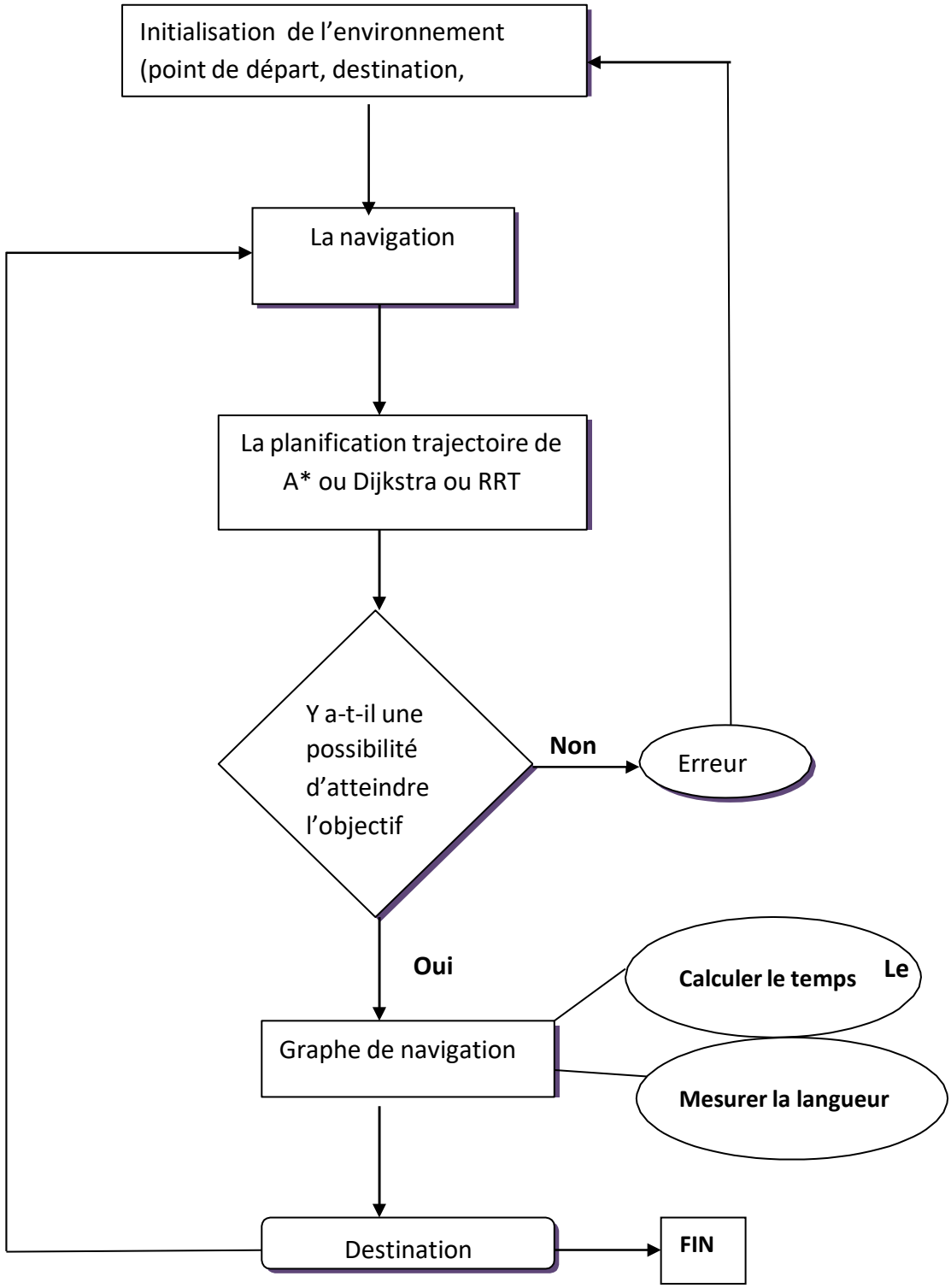


Figure3.2 : Organigramme de l'application.

2-2-3. Explication du schéma :

- **Initialisation de l'environnement :** Le robot est équipé de capteurs tels que des caméras, des capteurs de profondeur, des capteurs de proximité, etc. et d'autres éléments présents dans l'environnement du robot. Comme le montre l'image3.3 suivantes un exemple d'environnement simple d'obstacles.

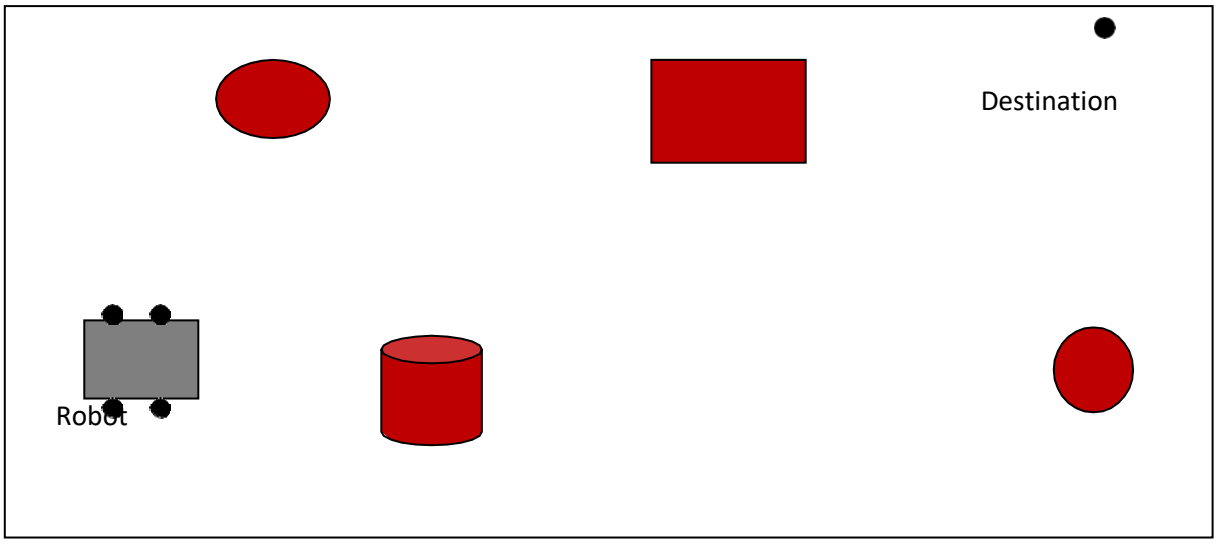


Figure3.3 Environnement simple d'obstacles.

- **La navigation :** Cela implique d'être conscient de l'environnement, de planifier les mouvements et d'effectuer les actions nécessaires pour atteindre une destination ou effectuer une tâche spécifique. Il existe plusieurs méthodes dont la planification d'itinéraire qui est notre sujet : Dans cette approche, le robot utilise une carte de l'environnement pour planifier un chemin vers sa destination. La figure3.4 illustre un exemple de navigation avec détection des obstacles.

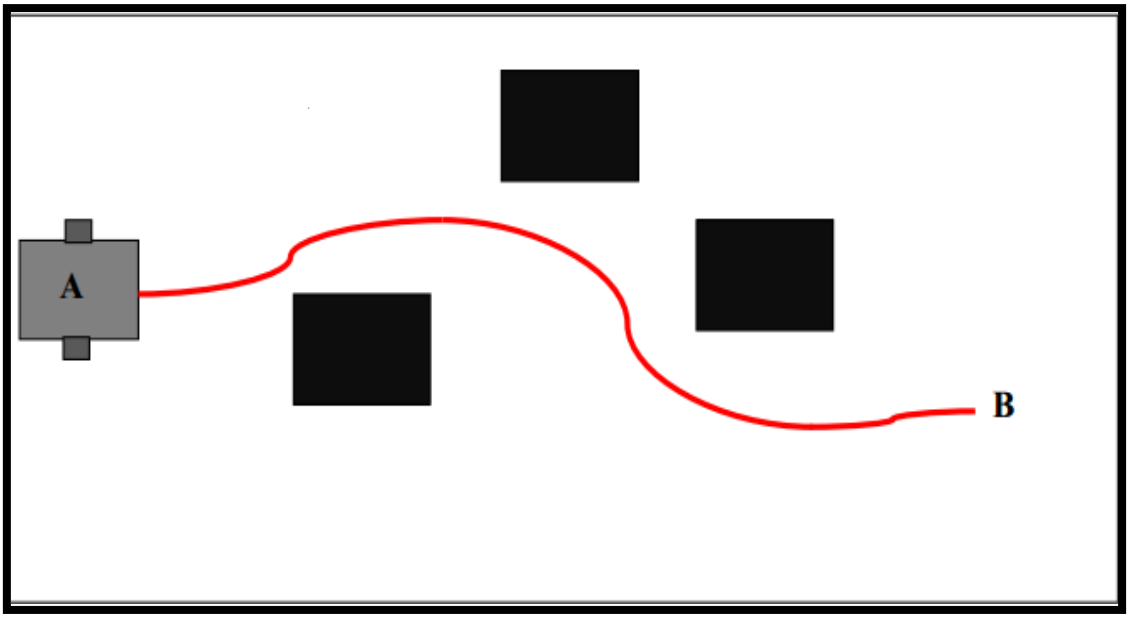


Figure3.4 : navigation avec détection des obstacles

Évitement d'obstacle :

L'évitement d'obstacles est un comportement de base que l'on retrouve dans presque tous les robots mobiles.

Le bot lui permet d'éviter les obstacles qui sont apparus dans son champ de vision, le bot doit donc sélectionner un autre chemin. Le comportement d'évitement d'obstacles est déterminé par un capteur en fonction de l'environnement dans lequel il évolue et de sa localisation par rapport aux obstacles rencontrés.

- **A*, RRT, DIJKSTRA:** La planification peut être basée sur des algorithmes de recherche, tels que l'algorithme A * (A-star), qui évalue différents chemins possibles en fonction de critères tels que la distance, les obstacles et les limitations du mouvement du robot, les algorithmes Dijkstra et RRT. Une fois qu'un itinéraire est planifié, le robot peut l'exécuter en contrôlant ses actionneurs.

La planification de trajectoire est le processus de détermination de la trajectoire optimale ou la plus courte entre deux points dans un environnement donné. Les algorithmes utilisés pour la planification de trajectoire diffèrent dans la manière dont ils atteignent cet objectif.

- Algorithme A* : L'algorithme A* est l'un des algorithmes les plus connus et efficaces pour la planification de trajectoire. Il se base sur la recherche de priorité et évalue le coût attendu pour atteindre la destination. A* utilise une fonction d'évaluation (appelée fonction de coût g) pour calculer le coût de transition d'un point à un autre, ainsi qu'une fonction d'estimation du coût attendu pour atteindre la destination à partir du point actuel (appelée fonction heuristique h). La priorité est calculée en utilisant la somme de ces deux fonctions (appelée fonction de coût f), et les points ayant le coût attendu le plus faible sont explorés en priorité. A* utilise des structures de données telles que la file de priorité (Priority Queue) pour organiser le processus d'exploration.

- Algorithme Dijkstra : L'algorithme Dijkstra est l'un des algorithmes de base pour la planification de trajectoire. Il se base également sur la recherche de priorité et calcule la plus courte distance possible entre le point de départ et tous les autres points dans l'environnement. Une valeur initiale de distance de départ vers le nœud courant est définie à 0, puis la distance est mise à jour pour tous les nœuds adjacents au nœud courant. Ce processus est répété jusqu'à ce que tous les nœuds possibles soient explorés et que la trajectoire optimale soit déterminée. Dijkstra utilise des structures de données telles que la file de priorité pour organiser les nœuds à explorer.

- Algorithme RRT : L'algorithme RRT (Rapidly-exploring Random Tree) est un algorithme d'échantillonnage utilisé pour la planification de trajectoire dans l'espace bi- ou tridimensionnel. RRT commence par le point de départ et construit un arbre qui évolue de manière aléatoire. L'étape clé de RRT consiste à choisir un point aléatoire dans l'espace et à trouver le point le plus proche dans l'arbre existant, puis à créer une connexion entre ces deux points. Ce processus est répété jusqu'à atteindre la destination. RRT présente des avantages dans le traitement d'environnements complexes et inconnus, et fournit des solutions acceptables en peu de temps.

En résumé, les algorithmes A*, Dijkstra et RRT diffèrent dans leur approche pour atteindre la planification optimale des trajectoires. A* et Dijkstra utilisent la recherche de priorité et améliorent les performances en établissant des priorités en fonction du coût, tandis que RRT se base sur l'échantillonnage aléatoire et explore rapidement l'espace pour trouver une trajectoire possible.

- **Y a-t-il une possibilité d'atteindre l'objectif:** C'est-à-dire que s'il n'y a pas de port entre le point de départ et le point cible dans l'espace de travail en raison d'obstacles, on passe à non, et on passe à oui s'il est possible de contourner les obstacles et d'atteindre l'objectif.

- **Graph de navigation :** Cela montre la forme finale résultant de la planification du chemin pour l'un des trois algorithmes pour que le robot atteigne l'objectif souhaité.

  Explication détaillée :

  La navigation globale d'un robot mobile fait référence à sa capacité à tracer et à suivre des chemins à travers un environnement complexe pour atteindre une destination spécifique. Cette tâche est généralement accomplie à l'aide de techniques telles que la cartographie de l'environnement, la localisation et la planification d'itinéraire.

Processus global de navigation d'un robot animatronique :

- Cartographie de l'environnement : le robot commence par explorer l'environnement dans lequel il se trouve et construit une carte de cet environnement. Cela peut être accompli à l'aide de capteurs tels que des caméras, des lidar (télémètres laser) ou des scanners de profondeur.

- Localisation : Une fois que le bot a une carte de l'environnement, il doit se situer par rapport à cette carte. Cela peut être fait en utilisant des techniques de localisation telles que la localisation et la cartographie simultanées (SLAM : Simultaneous Localisation And Mapping) ou des techniques basées sur des marqueurs ou des signaux spécifiques dans l'environnement.

Localisation du robot dans un environnement :

Dans cette partie nous parlerons de la localisation des bots. Lorsque nous avons une carte de notre environnement, il est impératif que notre bot sache à tout moment quelle est sa position et son orientation sur la carte. C'est extrêmement important, car si nous n'avions pas cette information, la carte serait complètement inutile.

Comme :

Allez dans une nouvelle ville et achetez une carte de cette ville, mais si nous ne savons pas où nous sommes sur la carte, cette dernière sera complètement inutile. Donc

La localisation du robot est essentielle.

- Planification d'itinéraire : Une fois que le robot connaît son emplacement sur la carte, il peut planifier un itinéraire pour atteindre sa destination. Cela implique de trouver le chemin optimal dans l'environnement tout en évitant les obstacles et en tenant compte des limites du robot, telles que sa vitesse de pointe ou sa maniabilité.

- Suivi de chemin : une fois qu'un chemin est planifié, le robot doit suivre ce chemin pour atteindre sa destination. Cela peut impliquer l'utilisation de capteurs tels que des odomètres, des centrales inertielles ou des systèmes de vision pour surveiller le mouvement du robot par rapport à la trajectoire prévue et apporter des corrections si nécessaire.

- **Destination** : est l'endroit où le robot veut aller, après avoir planifié son chemin pour atteindre cette destination. Une destination est un objectif quidoit être atteint.

3. Conclusion :

Dans ce chapitre, nous avons fourni des spécifications pour la conception de notre système. Cette étape représente l'une des étapes les plus importantes de notre processus de développement de système.

Décrit la conception de notre système d'un point de vue général et détaillé pour comprendre et réussir l'étape de programmation. Dans le chapitre suivant, nous expliquerons dans quelle mesure notre système a atteint certains des résultats obtenus, avec les structures qui ont été choisies pour mettre en œuvre ce système.

Chapitre04

Les resultats

1. Introduction :

Dans ce chapitre, nous expliquerons la mise en œuvre des différentes étapes de notre système conçu dans le chapitre précédent, et nous commencerons par vous dévoiler l'environnement de développement utilisé, puis les fonctions utilisées, et enfin nous présenterons quelques résultats expérimentaux de notre travail, en utilisant le système d'exploitation ROS (En anglais: Robot Operating System)qui permet le développement de bots.

2. ROS :

Un système d'exploitation de robot est un ensemble d'outils informatiques, un middleware robotique (c'est-à-dire un ensemble de Framework logiciels permettant de développer des logiciels de robots) développé en 2007 par la société américaine « Willow Garage », pour son sur www.openrobotics.org (consulté le 7 avril 2022) (anciennement Open Source Robotics Foundation, ou OSRF). ROS prend officiellement en charge plus de 75 robots [27]. Les principales bibliothèques clientes ROS (C++, Python et Lisp) sont orientées vers un système de type Unix, principalement parce qu'elles s'appuient sur de vastes collections de dépendances logicielles open source.

Description :

Le logiciel ROS peut être séparé en trois groupes :

- Les outils utilisés pour créer, lancer et distribuer des logiciels basés surROS (roscore [28], roslaunch [29], catkin [30])
- Les clients ROS pour des langages : roscpp (C++)[31] et rospy(python)[32]
- Les package contenant.

- Des programmes pour ROS utilisant un ou plusieurs clients ROS.

Il existe des clients ROS non officiels. Parmi eux on peut notamment citer :

- Rosjava (Java) [33] qui a permis de faire fonctionner ROS sur le système d'exploitation Androïde.

- Roslibjs (JavaScript) [34] qui permet d'interagir avec un système ROS depuis un navigateur. Ce client est développé dans le cadre du Robot Web Tools effort.

Fonctionnalités :

ROS regroupe plusieurs fonctionnalités qui facilitent le développement d'applications soupleset modulables pour la robotique :

- Architecture de communication inter-processus et inter-machine.
- Serveur de paramètre.
- Système de test.
- Simulateur (Gazebo [35]).

Architecture de communication :

ROS offre une architecture souple de communication inter-processus et inter-machine. Les processus ROS sont appelés des nodes et chaque node peut communiquer avec d'autres *via* des topics. La connexion entre les nodes est gérée par un *master* et suit le processus suivant :

- Un premier node avertit le master qu'il a une donnée à partager.
- Un deuxième node avertit le master qu'il souhaite avoir accès à une donnée.
- Une connexion entre les deux nodes est créée.

- Le premier node peut envoyer des données au second.

Un *node* qui publie des données est appelé un publisher et un *node* qui souscrit à des données est appelé un subscriber. Un *node* peut être à la fois publisher et subscribe.

Histoire:

ROS a été initialement développé en 2007 sous le nom switchyard par le Stanford Artificial Intelligence Laboratory .

De 2008 à 2013, le développement a été effectué principalement par Willow Garage, un institut / incubateur de recherche en robotique. Pendant ce temps, des chercheurs de plus de vingt institutions ont collaboré avec les ingénieurs de Willow Garage dans un modèle de développement fédéré [36],[37].

En février 2013, le développement de ROS est poursuivi par l'Open Source Robotics Foundation18. En août 2013, un blog annonce que "Willow Garage" sera absorbée par une autre société créée par son fondateur, Suitable Technologies19. Le support du robot PR2 créé par "Willow Garage" a été par la suite repris par Clearpath Robotics [38].

Historique des versions :

Les versions de ROS peuvent être incompatibles entre elles et sont souvent désignées par leur nom de code plutôt que leur numéro de version. La nouvelle version utilisée est: **ROS NOETIC.**

Distribution	Date de Publication	Image	Date de fin de vie
Noetic Ninjemys(dernier version de ROS 1)	23 mai 2020		30 mai 2025

Tableau4.1 : la dernière version de ros1

3. Implémentation :

Les outils de travail :

Figure4.1: Image de logicielle Visual studio code

Visual studio code : C'est un éditeur de script de Microsoft. L'éditeur est open source et fonctionne sur les systèmes d'exploitation Windows, Mac OS et Linux. L'éditeur est basé sur l'environnement Electron et a été publié par Microsoft en avril 2015.

Qu'est-ce que rviz ?

Rviz (abréviation de "visualisation ROS") est un outil logiciel de visualisation 3D pour les robots, les capteurs et les algorithmes. Il vous permet de voir la perception du robot de son monde En visualisant ce que le robot voit, pense et fait, l'utilisateur peut déboguer une application automatisée des entrées de capteur aux actions planifiées(ou non planifiées).

Rviz affiche les données des capteurs 3D pour les caméras stéréo, les lasers, la cinétique et d'autres appareils 3D sous la forme de nuages de points ou d'images de profondeur.

Les données des capteurs 2D des webcams, des caméras RVB et des télémètres laser 2D peuvent être affichées par Rviz sous forme d'image. Si un vrai robot entre en contact avec un poste de travail exécutant Rviz, ce dernier Il affichera la configuration actuelle du bot sur le modèle de bot par défaut. Par exemple, si les bras d'un vrai robot à deux bras comme Baxter sont dans une certaine position, le modèle de robot affichera cette position dans Rviz. Rviz fournit une interface utilisateur graphique (GUI) configurable pour permettre à l'utilisateur d'afficher uniquement les informations pertinentes pour la tâche en cours. [39]

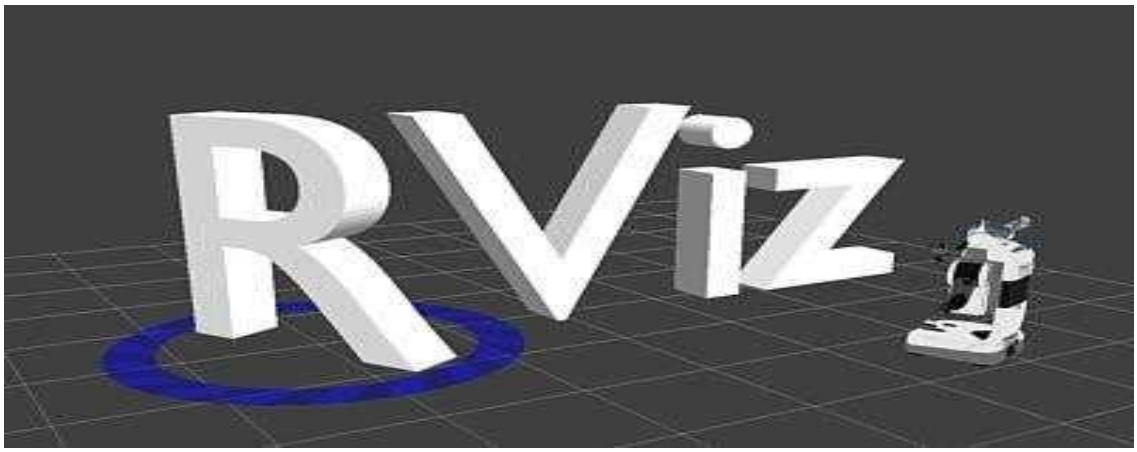


Figure4.2 : Image de logicielle Rviz.

C'est quoi Gazebo ?

Gazebo est un simulateur de robot 3D. Son objectif est de simuler un robot, vous donnant un substitut proche de la façon dont votre robot se comporterait dans un environnement physique réel.

Figure 4.3: Image de logicielle Gazebo.

L’espace de travail :

Il s'agit d'un espace équipé d'un robot et d'un ensemble d'obstacles, utilisant deux programmes de **gazebo** avec **rviz** afin d'effectuer l'opération comme indiqué dans la figure suivante :

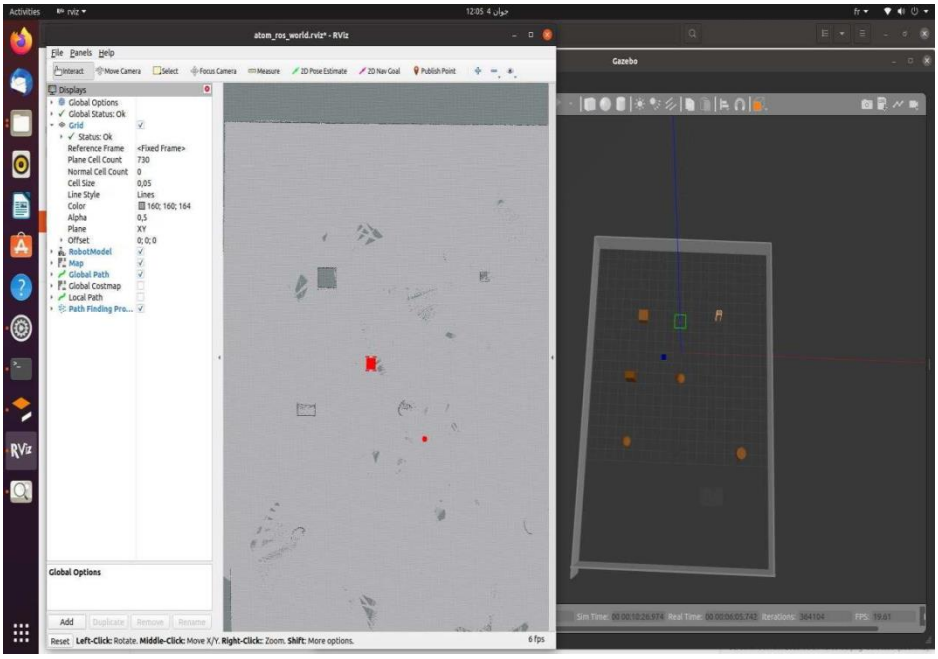


Figure 4.4 : l’environnement dans le GAZEBO/RVIZ.

L’algorithme Dijkstra:

L'algorithme de Dijkstra produit l'arbre du chemin le plus court à partir de la source donnée comme racine de cet arbre. Cet algorithme procède dans son travail avec deux groupes, le premier contient les sommets qui sont dans l'arbre des plus courts chemins, et le second contient les sommets qui n'ont pas encore été ajoutés à l'arbre des plus courts chemins. A chaque étape de l'algorithme, nous recherchons le sommet dans l'ensemble des sommets non ajoutés à l'arbre qui a la distance la plus courte possible de la source.

Fonctionnement de l'algorithme de Dijkstra :

L'algorithme de Dijkstra produit le chemin le plus court chemin en partant de la source donnée comme racine de ce chemin, puis cet algorithme procède dans son travail avec deux groupes, le premier contient les nœuds dans le chemin le plus court chemin, et le second contient les nœuds qui n'ont pas encore été ajoutés au chemin le plus court path . A chaque étape, l'algorithme recherche le nœud dans l'ensemble de nœuds qui n'est pas ajouté à l'arbre et qui a la distance la plus courte possible de la source jusqu'à ce que tous les nœuds existants soient traversés.

Le code de l'algorithme Dijkstra :

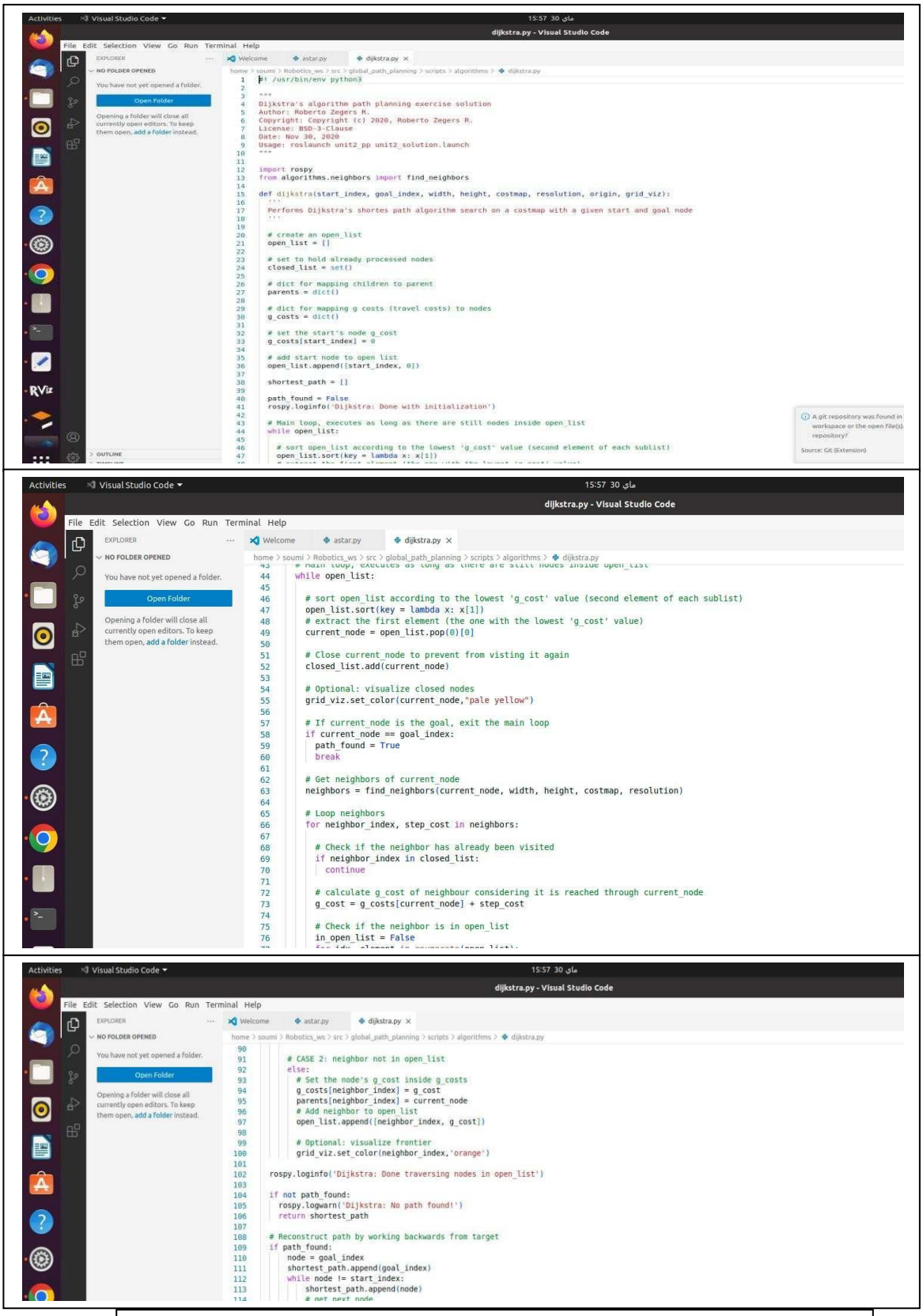


Figure4.5 le code de l'algorithme Dijkstra

Simulateur Androïde :

Ici nous voyons le chemin du bot par l'algorithme Dijkstra :

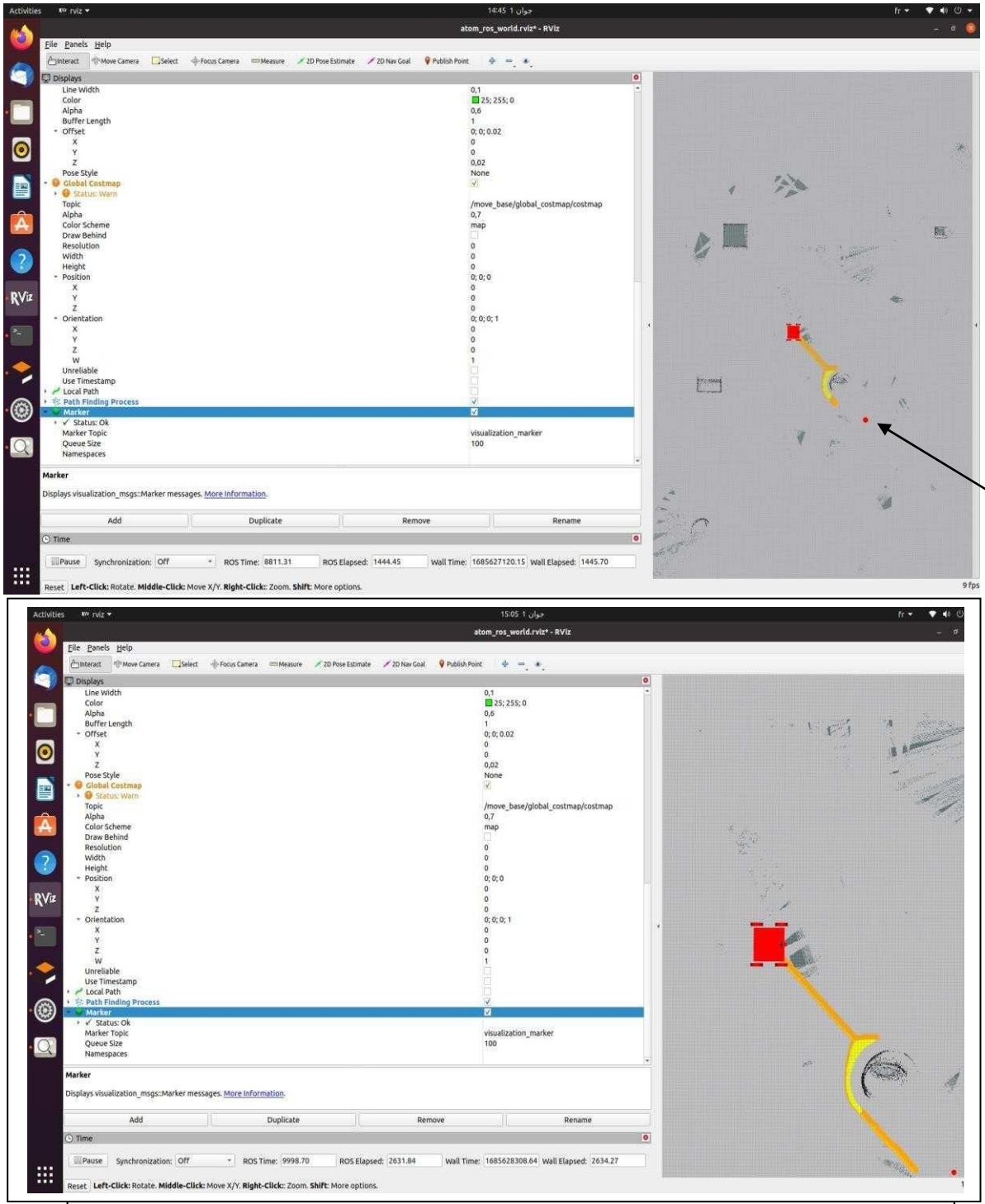


Figure4.6 : Le processus de planification du chemin avant d'atteindre le but

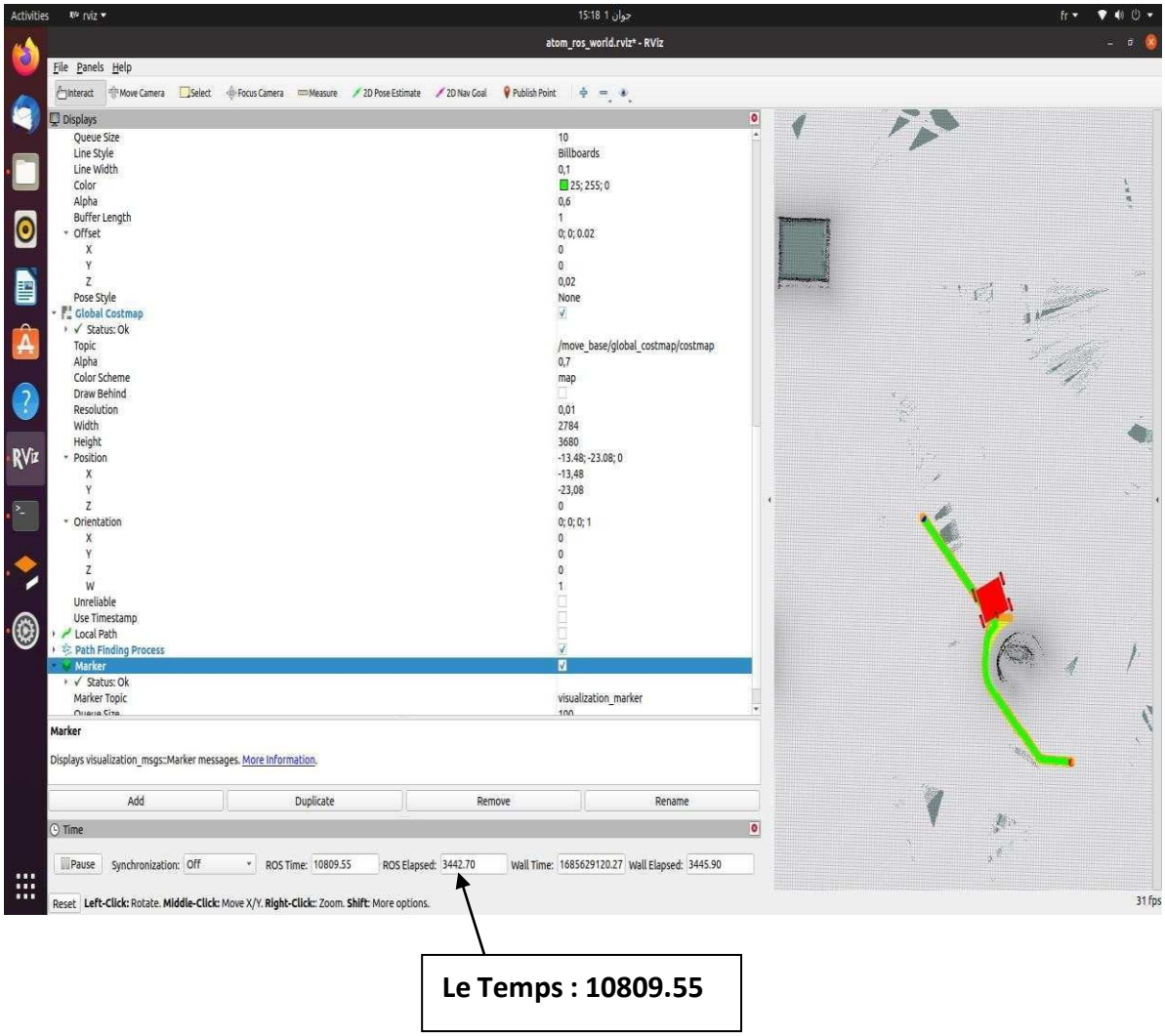


Figure4.7 : Trajectoire de planification du mouvement du robot vers la cible

L'algorithme A* :

L'algorithme A* résout les problèmes plus rapidement et plus efficacement.

Tous les graphiques ont différents nœuds ou points que l'algorithme doit prendre pour arriver au nœud final. Tous les chemins entre ces nœuds ont une valeur numérique, qui est le poids du chemin. La somme de tous les itinéraires traversés vous donne le coût de cet itinéraire.

Dans un premier temps, l'algorithme calcule le coût pour tous ses nœuds voisins et sélectionne le nœud le moins cher. Ce processus est répété jusqu'à ce qu'aucun nouveau

nœud ne puisse être sélectionné et que tous les chemins aient été parcourus. C'est un algorithme caractérisé par l'idéalisme et le perfectionnisme, et l'idéalisme signifie ici qu'il trouvera sûrement le chemin le moins cher (donne la solution optimale) et le perfectionnisme qu'il trouvera tous les chemins disponibles qui relient le point de départ et le stable.

L'algorithme de recherche A* utilise également une fonction heuristique qui fournit des informations supplémentaires sur la distance qui nous sépare d'un nœud par rapport à l'objectif.

Le code de l'algorithme A* :

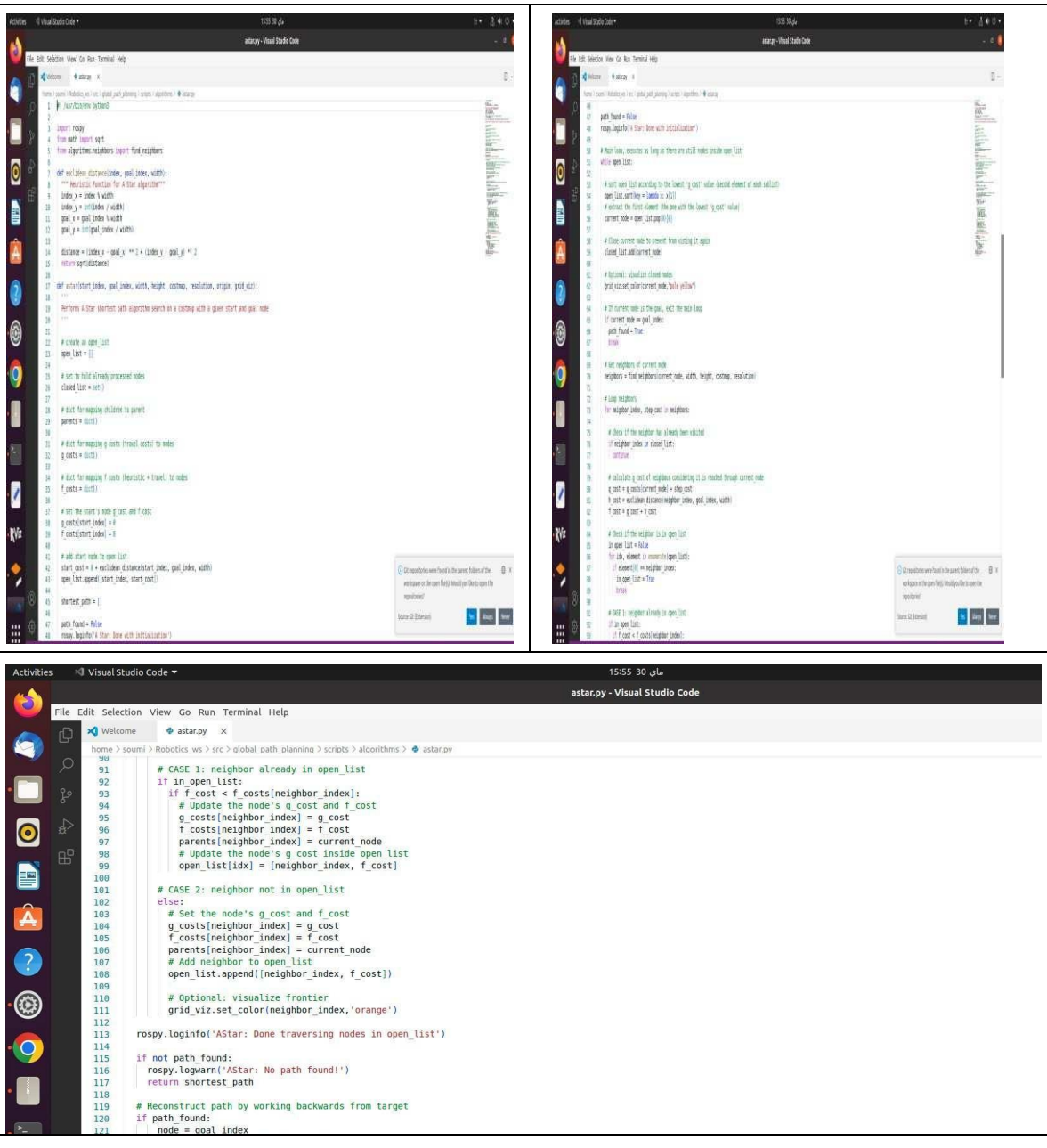


Figure4.8 Le code de l'algorithme A*

Simulateur Androïde

Ici nous voyons le chemin du bot par l'algorithme A* :

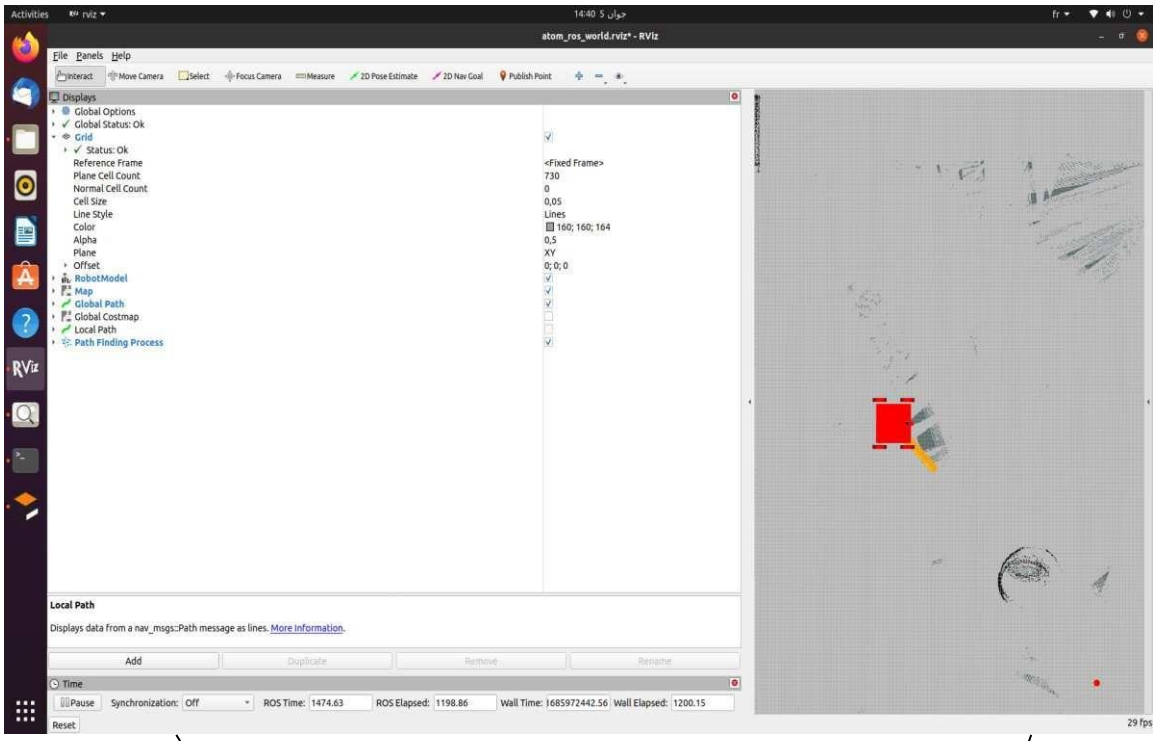


Figure4.9 : Début de la planification des pistes pour A*

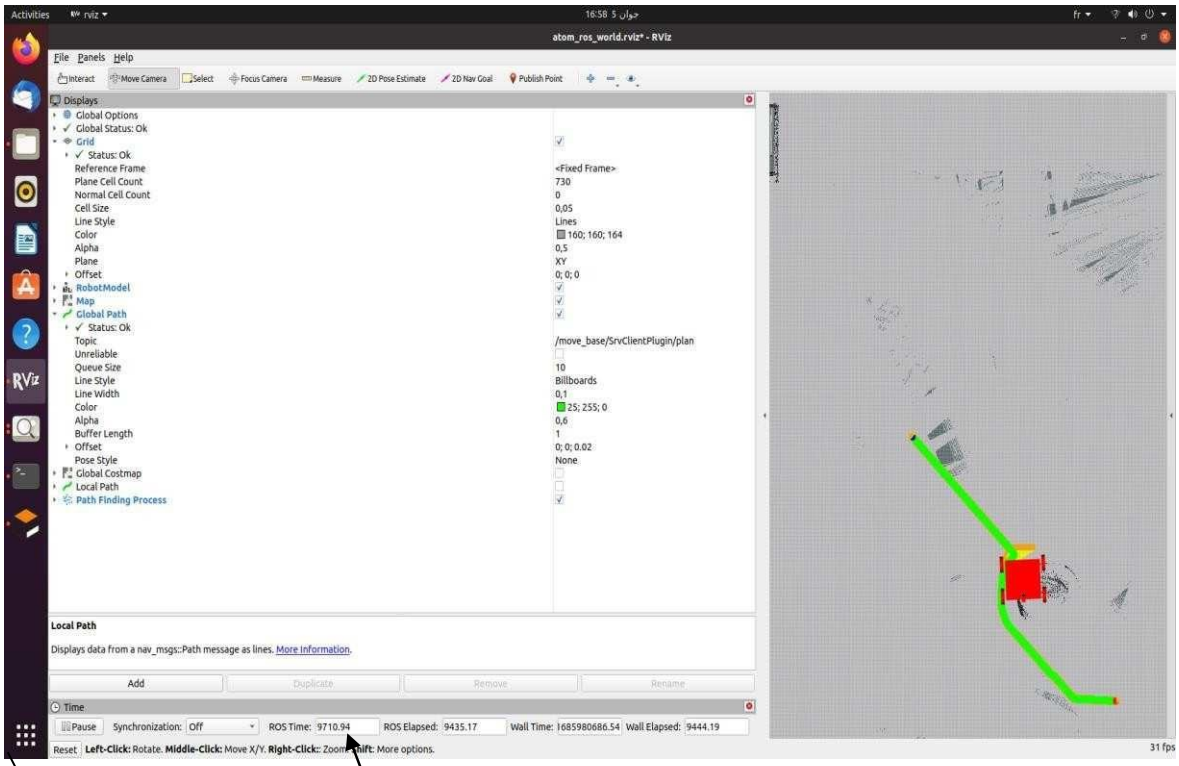


Figure4.10 : Fin de planification de trajectoire et déplacement du robot par A*.

Le Temps : 9710.94

L'algorithme RRT :

L'algorithme RRT possède plusieurs fonctionnalités qui le rendent utile pour planifier des trajets et résoudre des problèmes de mobilité. Voici quelques caractéristiques notables de l'algorithme RRT :

- Vitesse d'exploration : L'algorithme RRT vise à explorer rapidement l'espace en générant un arbre aléatoire. Le processus d'exploration signifie que l'algorithme peut couvrir une vaste zone en peu de temps et trouver rapidement une solution.

- Évolutivité et modification : L'algorithme RRT peut être facilement modifié pour s'adapter à un problème de planification spécifique. Diverses restrictions et restrictions peuvent être ajoutées au mouvement et préciser les limites et les conditions requises pour générer les chemins.

- Adaptabilité aux espaces de grande dimension : L'algorithme RRT gère bien les espaces de grande dimension, où les complexités spatiales et le nombre d'états possibles peuvent être très importants. Grâce à l'exploration aléatoire et à la génération de nœuds, l'algorithme peut relever ces défis.

- Capacité à gérer les obstacles et les zones complexes : L'algorithme RRT peut gérer les obstacles et les zones complexes dans l'espace. Lorsque l'algorithme génère un nouveau point dans l'arbre, il vérifie si le point est situé dans des obstacles ou des régions non valides. L'algorithme peut être modifié pour éviter ces obstacles ou pour générer des chemins pour les contourner.

- Accessibilité à plusieurs solutions

Le code de l'algorithme RRT :

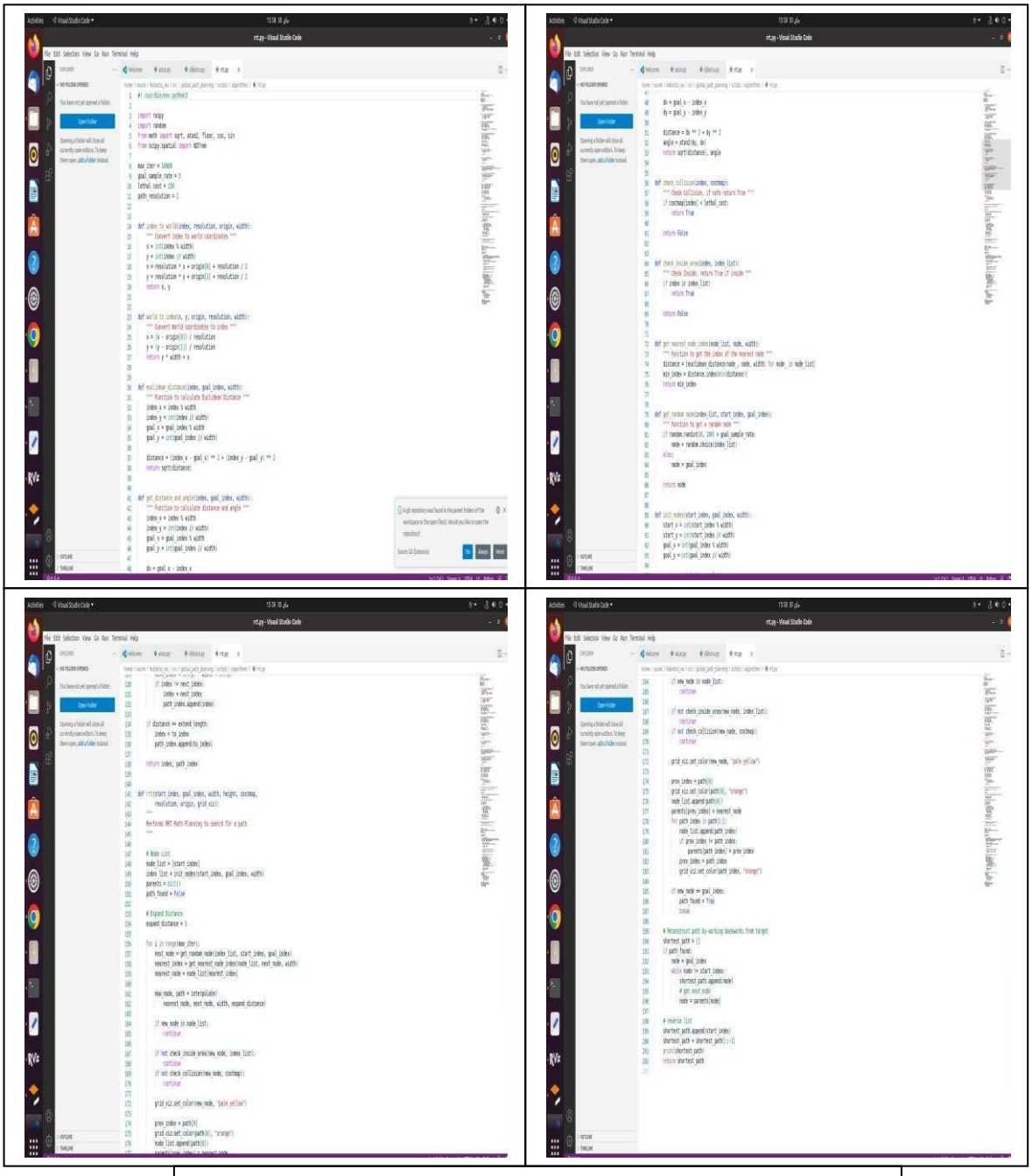


Figure4.11 : Le code da l'algorithme RRT

Simulateur Androïde :

Ici nous voyons le chemin du bot par l'algorithme RRT:

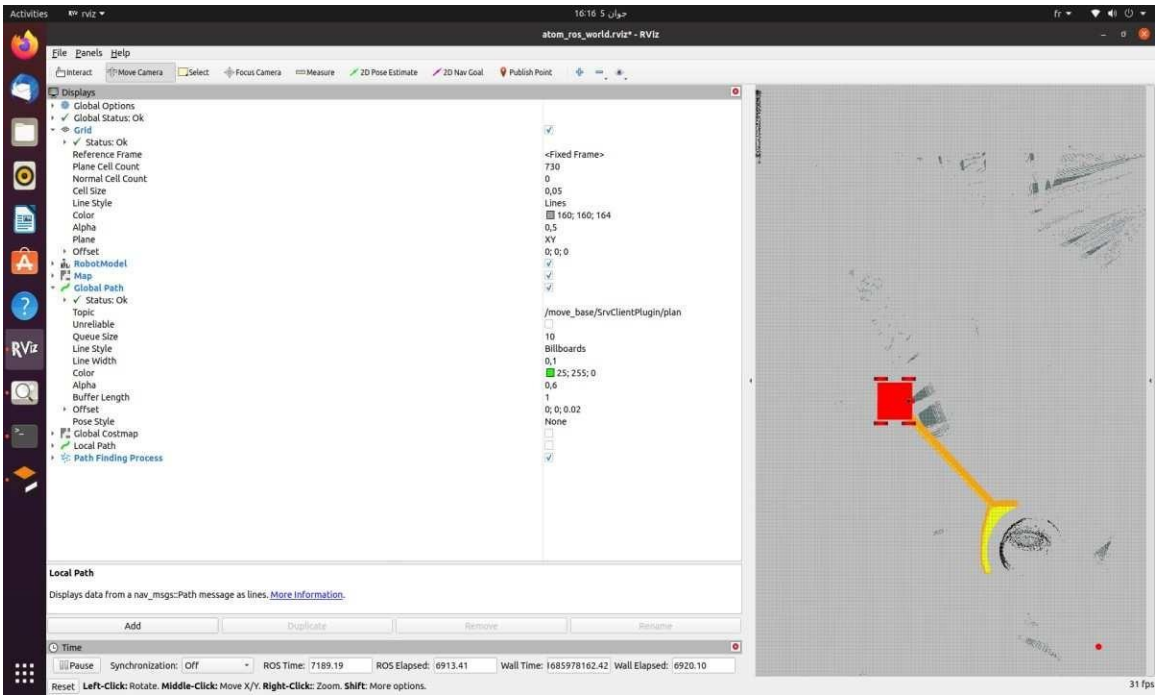


Figure 4.12: Pendant processus de planification de trajectoire d'un algorithme RRT.

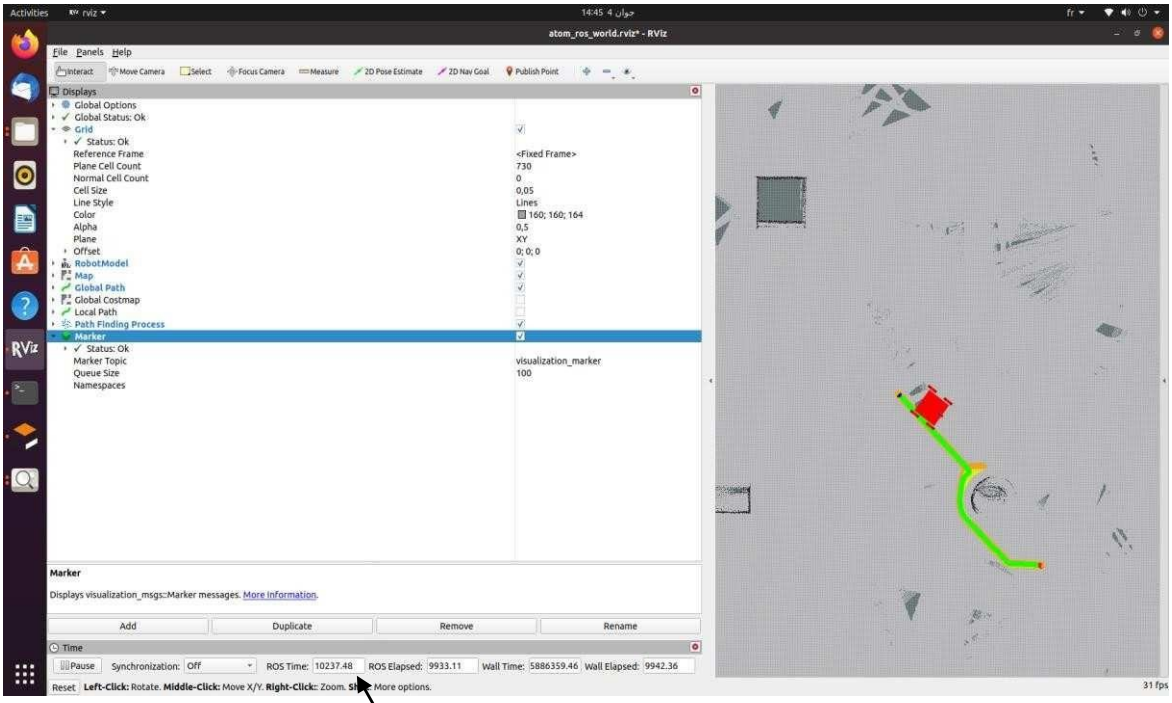


Figure4.13 : Planification d'itinéraire par RRT.

Le Temps : 10240.34

4. Conclusion :

Dans ce chapitre, nous avons expliqué comment planifier l'itinéraire pour les trois algorithmes de navigation globale, à savoir, A*, Dijkstra, et RRT.

Grâce aux résultats que nous avons obtenus, le classement des algorithmes peut être classé en fonction du temps passé pendant le processus de planification du chemin, où A * est le premier, le second est RRT et enfin Dijkstra.

Conclusion générale

Conclusion générale :

Le présent travail met en œuvre une étude comparative approfondie de ces trois algorithmes en utilisant différents critères d'évaluation tels que la qualité du chemin trouvé, le temps de calcul, la complexité algorithmique, la robustesse face aux changements d'environnement, etc. Les performances des algorithmes sont mesurées en utilisant des scénarios de navigation complexes et des métriques appropriées.

Les résultats de l'étude comparative sont analysés et discutés, mettant en évidence les forces et les faiblesses de chaque algorithme. Ces résultats permettent de mieux comprendre les avantages et les limitations des algorithmes A*, Dijkstra et RRT dans le contexte de la navigation globale des robots autonomes.

En conclusion, cette étude comparative fournit des informations précieuses pour les concepteurs de systèmes de navigation pour robots autonomes, en les aidant à choisir l'algorithme le plus adapté à leurs besoins spécifiques en fonction des contraintes de leur environnement et des objectifs de leur application.

Référence:

[1] :W.S Hao,"Animating Autonomous Pedestrians ",Thèse de Doctorat, Institude Mathematical Sciences ,Université de New York 2006.

[2]: Définition surle site de l'ATILF, www.atilf.fr., ressources-grand public.

[3] :<<C dans l'air :les robots sont parmi nous>>[archive] un reportage dans France5,15/03/2012.

[4] :(en)BarmesD.P,Summers,P,Shshaw,A,Aninvestiation.

[5] :(en) A.Colozza,G.ALandis,andV.Lyons,NASA TM-2003-212459(july 2003).

[6] :(en)<<RoboCourierAutonomous Mobile Robot>>[archive],sur Swisslog_inspiredsoulusion(consultéle 5 mai 15)

[7] : Levitt TS. Mobilité qualitative des robots mobiles. Journal international de l'intelligenceartificielle.1990 ;44(3) : 305-360.

[8] :DT 436 at :http://www.researchgate.net/publication/332717157.

[8] :Chos et, Howei, Burdick, and Joel, "Sensor-based exploration: The hierarchical generalized Voronoigraph," The International Journal of Robotics Research, vol. 19, no. 2, pp. 96-125, 2000.

[9] :L. Lulu and A. Elnagar, "A comparative study between visibility-based roadmap path planning algorithms," In : 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.3263-3268, 2005.

[10] :M. D. Berg, M. V. Kreveld, M. M. Overmars and O. C. Schwarzkopf, "Computational geometry,"Springer Berlin Heidelberg, vol. 1-17, 2000.

[11-13]:H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavriki and S. Thrun, "Principles of Robot Motion," MIT Press, Cambridge, MA, 2005.

O. Takahashi and R. J. Schilling, "Motion planning in a plane using generalized Voronoidiagrams," IEEE Transactions on Robotics and Automation, vol. 5, no. 2, pp. 143-150, 1989.

C. O. Dunlaing and C. K. Yap, "A retraction method for planning the motion of a disc," Journal ofAlgorithm, vol. 6, pp. 104-111, 1985.

[14] :E. Masehian and M. R. Amin-Naseri, “A Voronoi diagram-visibility graph-potential field compoundalgorithm for robot path planning,” Journal of Robotic System, vol. 21, pp. 275-300, 2004.

[15] :G. Sanchez and J. Latombe, “A Single-query Bidirectional Probabilistic Roadmap Planner with LazyCollision Checking,” Springer Tracts in Advanced Robotics, vol. 6, pp. 403-417, 2001.

[16] : F. Yan, Y.S. Liu, and J.Z. Xiao, “Path planning in complex 3D environments using a probabilisticroadmap method”, International Journal of Automation and computing, 10(6), pp.525-533, 2013.

[17] :O. Khatib, “Real time Obstacle Avoidance for manipulators and Mobile Robots,” IEEE InternationalConference on Robotics and Automation, Missouri, vol. 25-28, pp. 500-505, Mar 1985.

[18] : O. Khatib, “Real time Obstacle Avoidance for manipulators and Mobile Robots,” IEEE International Conference on Robotics and Automation, Missouri, vol. 25-28, pp. 500-505, Mar 1985

[19] :K. P. Valavanis, T. Hebert, R. Kolluru and N. Tsourveloudis, “Mobile robot navigation in 2D dynamic environments using an electrostatic potential field,” IEEE Transactions on Systems Man andCybernetics, Part A: Systems and Humans, vol. 30, no. 2, pp. 187-196, 2000.

[20] :R. Brooks, "A Robust Layered Control System for a Mobile Robot", IEEE Journal ofRobotics and Automation, vol. 2, no.1, pp.14-23, 1986.

[21]:F. Hoffman, "Soft Computing Techniques for the Design of Mobile Robot Behaviors",Information Sciences, vol. 122, pp. 241-258, 2000.

[22]:A. Fatmi, A. Al Yahmadi, L. Khriji, and N. Masmoudi, "A Fuzzy Logic Based Navigation of a Mobile Robot", World Academy of Science, Engineering and Technology, vol. 22, pp. 169-174, 2006.

[23]:D. Filliat, "Robotique Mobile", Cours à l'école Nationale Supérieur des TechniquesAvancées ENSTA, Octobre 2004.

[24]:S. G. Shuzhi, F. L. Lewis, "Autonomous Mobile Robots, Sensing, Control, Decision,Making and Applications", Taylor and Francis Group, 2006.

[25] :Perle, Judée (1984). Heuristique : stratégies de recherche intelligentes pour la résolution de problèmes informatiques. Addison-Wesley. ISBN 0-201-05594-5. Archivé de l'original le 18 mai 2020.

[26] :ARA *: Recherche A * à tout moment avec des limites prouvables sur la sous-optimalité ". Dans S.Thrun, L. Saul et B. Schölkopf, éditeurs, Actes de la Conférence sur les systèmes de traitement de l'information neuronale (NIPS), Cambridge, MA, 2003. Presse du MIT.

Références bibliographiques

[27] :Zeng, W.; Church, RL (2009). « Rechercher les chemins les plus courts sur des réseaux routiers réels : le cas de A* ». Journal international des sciences de l'information géographique. c. 23 p. 4 :531–543. doi:10.1080/13658810801949850. Archivé de l'original le 8 octobre 2018.

[28] :Community Metrics Report 2015 -http://download.ros.org/downloads/metrics/metrics-report-2015-07.pdf [archive]

[29]:« *roscore - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023) [30]:« *roslaunch - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023)[31]:« *catkin - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023) [32]:« *roscpp - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023) [33]:« *roysp - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023)) [34]:« *rosjava - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023) [35]: « *roslibjs - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023)

[36]: « *gazebo_ros_pkgs - ROS Wiki* » [archive], sur wiki.ros.org (consulté le 12 juin 2023)[37]:« *ROS Wiki* » [archive],sur www.ros.org (consulté le 12 juin 2023)

[38] :Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, RobWheeler, Andrew Ng.http://www.robotics.stanford.edu/~ang/papers/icraoss09-ROS.pdf [archive]

[39] : (en-US) « *Clearpath Welcomes PR2 to the Family - Clearpath Robotics* » [archive],sur ClearpathRobotics (consulté le 12 juin 2023)

[40]:Carol Fairchild et Thomas L Harman. ROS Robotics By Example: Learning to control wheeled, limbed, and flying robots using ROS Kinetic Kame. Packt PublishingLtd, 20