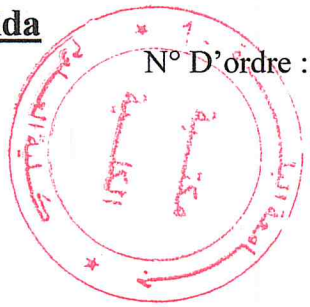
République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab Blida

N° D'ordre :.....



Faculté des sciences

Département d'informatique

Mémoire Présenté par :

Kaid-youcef Nesrine et Hebib Djamila

En vue d'obtenir le diplôme de master

Domaine : Mathématique et informatique

Filière : Informatique
Spécialité : Informatique
Option : Ingénierie de logiciel

Sujet : Construction collaborative de carte par un groupe de robots mobiles

Soutenu le : 24/06/2015

M.Président :	Hammouda
M. Examinateur :	Sidemmou
M. Examinateur :	Zayer
Mme.	BenblidiaNadjia
Mr.	HentoutAbd el fatah

Promotion
2014 / 2015

Dédicaces Nesrine

Que ce travail témoigne de mes respects :

A mes parents Rachid et Houria

Grâce à leurs tendres encouragements et leurs grands sacrifices, ils ont pu créer le climat affectueux et propice à la poursuite de mes études.

Aucune dédicace ne pourrait exprimer mon respect, ma considération et mes profonds sentiments envers eux. Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

A ma sœur Lylia et à mon frère Samy.

A mes petits neveux.

A Mes grands parents, ainsi qu'à toute ma famille

Ils vont trouver ici l'expression de mes sentiments de respect et de reconnaissance pour le soutien qu'ils n'ont cessé de me porter.

A tous mes professeurs

Leur générosité et leur soutien m'oblige de leurs témoigner mon profond respect et ma loyale considération.

A mes amis : Sonia , ahlem , Djamila, Meriem et Dahmen

Ils vont trouver ici le témoignage d'une fidélité et d'une amitié infinie.

Dédicaces Djamila

Que ce travail témoigne de mes respects :

A mes parents Abdesslam et Bahia

Grâce à leurs tendres encouragements et leurs grands sacrifices, ils ont pu créer le climat affectueux et propice à la poursuite de mes études.

Aucune dédicace ne pourrait exprimer mon respect, ma considération et mes profonds sentiments envers eux. Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

A ma sœur Dihia et à mes frères Lounis et Abdennour.

A mes petits neveux.

A Mes grands parents, ainsi qu'à toute ma famille

Ils vont trouver ici l'expression de mes sentiments de respect et de reconnaissance pour le soutien qu'ils n'ont cessé de me porter.

A tous mes professeurs

Leur générosité et leur soutien m'oblige de leurs témoigner mon profond respect et ma loyale considération.

Remerciements

Tout d'abord Merci à DIEU Allah de nous avoir donné la volonté et le courage de mener ce travail jusqu'à la fin.

On tient à remercier notre encadreur Abd el Fateh HENTOUT de nous avoir proposé ce sujet de recherche, et d'être toujours montré à l'écoute tout au long de la réalisation de ce mémoire.

Nos remerciements s'adressent également à Mme Benblidia Nadjia, promotrice de ce mémoire,

On remercie nos parents pour leur contribution, leurs soutiens et leur patience

On remercie également nos familles, nos amis, en particulier Fouad qui nous a apporté beaucoup d'aide tout au long de la réalisation de ce travail

Merci a nos professeurs et enseignants d'avoir été la, de nous avoir énormément appris par la qualité des enseignements qu'ils nous ont prodigués.

الملخص

من المهام الهامة في مجال البحث الروبوتات هي أن تسمح لمجموعة من الروبوتات المتحركة لاستكشاف تعاونية و تعيين بيئة غير معروفة. استكشاف ورسم الخرائط الروبوتية لديها العديد من التطبيقات ،على سبيل المثال تحقيق النقاط التيقدتكون خطيرة جدا بالنسبة للبشر،مثل البحث و الإنقاذ بعد بعض الكوارث,

نقدم في مذكرة التخرج هذه حل لمشكلة رسم الخرائط متعددة الروبوت .يركز عملنا على استراتيجيات التوزيع للسماح بإكتشاف و رسم المحيط في أقصر وقت ممكن لذلك اخترنا بنية مركزية من شأنها توفير نظرة شاملة عن البيئة بأكملها.

بالإضافة إلى ذلك، إستعملنا خريطة التمثيل متري لإكتشاف كل المساحة لأنه يسمح تقديم التمثيل الهندسي للعالم بشكل واضح. لتنفيذ الحل عملنا مع جهاز Player/Stage وهذا يسمح لاختبار وتطوير الخوارزميات دون الحاجة إلى بيئة حقيقية. كما يوفر أدوات مفتوحة المصدر التي تبسط تطوير وحدة التحكم، وخاصة أنظمة متعددة الروبوت

في نفس الوقت مجموعة الروبوت المحمولة تكتشف البيئة المحلية وتسترد كافة البيانات المجاورة. يرسلها إلى بطاقات الخادم المركزي للإندماج في خريطة عالمية واحدة.

كلمات البحث: كلمات البحث: رسم الخرائط، والبناء التعاوني، أنظمة متعددة الروبوت المتنقلة، والهندسة المعمارية المركزية والخرائط الهندسية.

Résumé

L'une des tâches importantes dans la recherche en robotique est de permettre à un groupe de robots mobiles d'explorer et de cartographier de façon coopérative un environnement inconnu. L'exploration et la cartographie robotique ont de nombreuses applications, comme la recherche et le sauvetage après certaines catastrophes.

Nous présentons dans ce mémoire une solution à la problématique de la cartographie multi-robots. Notre travail se concentre sur des stratégies de déploiement afin de permettre d'explorer et de cartographier l'environnent en un temps minimal. Nous avons, pour cela, opté pour une architecture centralisée qui offre une vue globale de tout l'environnement. De plus, nous avons considéré une carte métrique pour la représentation de tout l'espace exploré, car elle apporte une représentation géométrique du monde de manière explicite.

Afin d'implémenter notre solution, nous avons travaillé avec le simulateur Player/Stage. Ce dernier permet de tester et de développer des algorithmes sans la nécessité d'un environnement réel. Il fournit également des outils open source qui simplifient le développement de contrôleur, en particulier les systèmes multi-robots.

Au fur et à mesure qu'un robot mobile du groupe explore son environnement local et récupère toutes les données de son voisinage, il les envoie au serveur central de cartes afin de les intégrer dans une seule carte globale.

Mots clés : Mapping, Construction collaborative, Systèmes multi-robots mobiles, Architecture centralisée, Cartes géométriques.

Abstract

One of the important tasks in robotics research is to allow a group of mobile robots to cooperatively explore and map an unknown environment. The exploration and robotic mapping have many applications, such as search and rescue after some disasters.

We present in this paper a solution for the problem of multi-robot mapping. Our work focuses on deployment strategies, to explore and map the environment in a minimum time. For that, we have opted for a centralized architecture for the mapping process which offers a global view of the entire environment. Then, we considered a metric card for the representation of the whole explored space because it provides a geometric representation of the world explicitly.

In order to realize our objective, we worked with the simulator Player / Stage because it can test and develop algorithms without the need for a real environment, and finally, it also provides Open Source tools that simplify the development of controller, especially multi-robot systems.

While a mobile robot of the group explores its local environment and collects all the data of its neighborhood, it sends them to the central server of maps in order to integrate them in a single global map.

Keywords: Mapping, Collaborative Construction, Multi-robot systems, Centralized architecture, Geometric cards.

Table des matières

Chapitre 2 : conception et implémentation

Chapitre 3 : tests et validation

Listes des figures

Chapitre 1 : Approches de construction collaborative de cartes

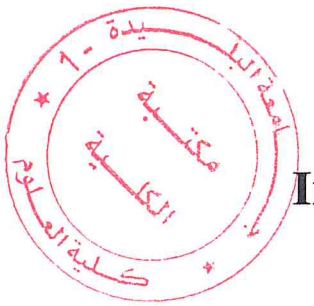
Chapitre 2 : Conception et implémentation

Chapitre 3 : Tests et validation

Liste des tableaux

Chapitre 3 : tests et validation

Introduction générale



Introduction générale

La robotique mobile est devenue une science importante qui implique de nombreuses thématiques telles que la mécanique, l'électronique, l'automatique, l'informatique et l'intelligence artificielle. Les avancées technologiques dans tous ces domaines ouvrent aujourd'hui aux chercheurs de nouvelles perspectives et des centaines de voies de recherche à explorer. Les résultats de recherche ont donné naissances à des systèmes multi-robots qui ont remplacé le système mono-robot afin de profiter de leurs avantages qui sont multiples [1].

Un système mono-robot ou Single-Robot System (SRS) a un seul robot qui possède la compétence de modéliser l'environnement. Il est conçu pour faire un travail seul. De telle façon les SRS sont des systèmes complexes et coûteux. Malgré qu'un robot possède un rendement élevé, il existe quelques tâches qui sont difficiles, et qui ne peuvent pas être effectuées comme des tâches séparées dans l'espace. Donc, il y aune restriction associée aux systèmes mono-robot, qui est la limitation dans l'espace [2].

Le concept de Systèmes Multi-Robots ou Multi-Robot Systems (MRS) débute dans les années 1990.Les MRS se réfèrent à des systèmes composés de plusieurs robots, homogènes ou hétérogènes, mobiles ou fixes. Ces robots sont capables de communiquer entre eux et coopérer pour améliorer l'efficacité d'exécution des tâches [2] [1].

Un système multi-robots peut effectuer des tâches difficiles ou même impossibles à accomplir par un seul robot. Une équipe de robots fournit une certaine redondance ; elle contribue à l'accomplissement d'une tâche de manière collaborative [1]. Cette collaboration devrait être en mesure d'apporter plusieurs avantages et intérêts par rapport aux SRS [2] [3] [4] :

- **Tolérance aux pannes :** Ceci se réfère à la notion de robustesse qui résulte de la fusion de données et du partage d'informations entre les robots. Par exemple, si un ou plusieurs robots produisent des fautes ou tombent en panne, cela n'entrainera pas l'arrêt systématique de la tâche ; les autres groupes peuvent continuer de fonctionner et aussi les remplacer pour accomplir leurs tâches.

- **Minimisation des coûts :** L'utilisation d'un groupe de robots peut s'avérer une solution plus simple et moins chère que la conception d'un seul robot dont la structure et le contrôle peuvent vite devenir complexe et encombrants.
- **Meilleures efficacités :** Les MRS produisent de meilleur résultat d'exécution comme la réduction du temps total d'exécution et l'économie de la consommation énergétique. Il peut aussi rendre une tâche possible, alors qu'elle peut être irréalisable par un seul robot. Par exemple, un robot unique peut ne pas avoir la force nécessaire pour pousser un objet lourd. L'utilisation d'un groupe de robots permet alors de diviser cette force sur les différentes entités robotiques qui peuvent appliquer leurs efforts et pousser ensemble dans la direction du mouvement souhaitée.
- **Amélioration de l'ensemble de tâches :** Un groupe de robots peut être en mesure d'accomplir de nouvelles tâches ou encore de perfectionner les tâches déjà accomplies.

La robotique mobile peut être utile dans presque tous les domaines ; les MRS peuvent apporter des nouvelles possibilités intéressantes par rapport aux solutions plus conventionnelles à un seul robot. Parmi les exemples d'applications de tels systèmes, nous pouvons penser aux suivantes : le sauvetage(la recherche des victimes à l'intérieur ou à l'extérieur des bâtiments) [5], le transport de gros objets [1], l'exploitation spatiale et sous-marine [6], la détection de feu forestier [7], l'inspection des endroits à risque (grotte, mines, sous-marin, aérien). Des possibilités d'applications sont également à prévoir dans d'autres secteurs comme l'industrie agricole et celle des transports.

Il existe plusieurs méthodes d'exploration pour la robotique mobile. Les plus simples sont des méthodes réactives qui permettent au robot de se déplacer aléatoirement ou de suivre une cible. Cependant pour accomplir une tâche complexe, il est souvent nécessaire de connaître la position du robot dans son environnement et de disposer d'une carte qui permet de planifier les déplacements pour atteindre un but précis en évitant les obstacles connus. Le robot doit notamment être capable de construire sa propre représentation de l'environnement. Cette tâche est nommée la cartographie [8].

Ces dernières années, il y a eu un intérêt significatif pour de divers aspects de la construction des cartes en robotique (c.-à-d. le mapping), de la cartographie pour un simple robot à la cartographie fortement distribuée en multi-robots.

Beaucoup de recherches ont été consacrée à la cartographie mono-robot. Cependant, il ya des problèmes qui sont difficiles à résoudre, tels que les erreurs de mesure, et la haute

dimensionnalité de la cartographie. L'utilisation de plusieurs robots pour la cartographie peut résoudre certains de ces problèmes et offrir aussi plusieurs autres avantages par rapport à la cartographie mono-robot. Nous citons, dans ce qui suit, les plus importants [9] :

- Souvent, il est possible de créer plus rapidement une carte de l'environnement avec plusieurs robots qu'avec un seul.
- La cartographie avec plusieurs robots est plus tolérante aux pannes que les plateformes mono-robot, c'est-à-dire la défection d'un robot ne signifie pas nécessairement la fin du processus de cartographie.
- Plusieurs robots peuvent également utiliser différents capteurs pour la cartographie, améliorant ainsi la qualité de la carte résultante.

L'intérêt de la cartographie est de définir des zones d'intérêt à explorer, des zones dangereuses à éviter ou encore trouver les zones navigables. Cette carte pourra servir de modèle de l'environnement à un opérateur humain qui voudrait intervenir dans une zone a priori inconnue. Elle pourra aussi ensuite être utilisée par d'autres robots pour y effectuer d'autres tâches ou servir aux opérateurs humains pour des environnements inaccessibles ou dangereux [8].

L'exploration et la cartographie d'environnement inconnu peut être utile pour les environnements lointains tels que la cartographie des planètes (Mars, Jupiter, etc.) ou encore pour la recherche et le secours après certaines catastrophes naturelles ou industrielles. Aussi, les applications robotiques de l'exploration et la cartographie sont nombreuses telles que la cartographie des villes, des bâtiments industriels ou des maisons.

L'objectif de ce projet est de développer une approche collaborative multi-robots pour une exploration complète d'un environnement quelconque. Afin de construire un modèle cohérent de l'environnement, les données recueillies par les différents robots doivent être intégrés dans une seule carte globale. Pour atteindre cet objectif, nous tenons à résoudre ces deux sous-problèmes :

- **Stratégie de déploiement :** L'exploration d'un environnement inconnu par plusieurs robots nécessite l'élaboration d'une stratégie de déploiement. Cette stratégie vise à réduire considérablement le temps nécessaire pour cartographier un environnement donné, car les robots peuvent explorer différentes parties en parallèle.
- **Stratégie de construction d'une carte :** La cartographie robotique collaborative consiste à représenter un environnement physique par un groupe de robots mobiles.

Ces derniers doivent être équipés de capteurs qui leurs permettent de percevoir le monde qui les entoure. Dans le but d'accomplir cette tâche, les robots doivent être capables de collecter les données nécessaires qui vont être intégrées et fusionnées dans une seule carte globale de représentation.

Dans ce mémoire nous présentons un système d'exploration et de cartographie de plusieurs robots. L'approche permet à cette équipe de robots mobiles de construire une carte globale précise d'un environnement d'intérieur inconnu, même quand les emplacements initiaux des robots sont inconnus.

À cet effet, outre cette introduction générale et une conclusion générale, nous avons décomposé ce mémoire en trois chapitres :

- Dans le premier chapitre, nous présentons les différentes approches de construction collaboratives, les différents objectifs et les travaux les plus récents et réputés dans le domaine de l'exploration et de la cartographie multi-robots. Nous présentons les différentes architectures et représentations de l'environnement.
- Dans le deuxième chapitre, nous introduisons notre travail de l'exploration et de la cartographie multi-robots. Ce chapitre est divisé en deux parties :
    - La première effectue une étude conceptuelle qui se présente sous forme de diagrammes UML.
    - La seconde partie décrit l'implémentation qui se base sur une méthode de détection d'obstacles et de cartographie.
- Dans le troisième chapitre, nous évaluons en simulation les performances de l'algorithme décrit dans le deuxième chapitre via plusieurs scénarios. Nous donnons pour chaque scénario le temps nécessaire pour la construction de la carte globale ainsi que la précision de cette dernière.

Chapitre 1

Approches de construction collaborative de cartes

Chapitre 1
Approches de construction collaborative de cartes

1. Introduction

L'exploration consiste à découvrir l'environnement avec un ou plusieurs capteurs montés sur des robots. Les robots ne doivent pas couvrir physiquement tout l'environnement mais doivent se déplacer afin que leurs capteurs puissent observer la totalité de l'environnement et, ainsi, construire une carte complète.

Générer des cartes est l'une des tâches fondamentales de robots mobiles. De nombreux systèmes robotiques réussis utilisent des cartes de l'environnement pour accomplir leurs tâches.

Pour une grande efficacité, de bonnes stratégies d'exploration sont exigées. En particulier, les robots ont besoin de savoir comment se répartir efficacement afin de cartographier toutes les zones inconnues.

Il existe plusieurs avantages en matière de cartographie avec plusieurs robots. La plus évidente est que plusieurs robots peuvent souvent faire le travail en moins de temps. Il est important en stratégies d'exploration de garder les robots relativement bien déployés et bien séparés.

2. Objectif de construction collaborative

L'exploration d'un environnement inconnu comporte différents objectifs selon les applications [10] :

- Tâche de recherche d'objets quelconque : l'exploration se base sur les plus grandes zones offrant une meilleure couverture afin de trouver les objets le plus rapidement possible.
- Tâche de surveillance : il faut faire une exploration totale de l'environnement en plusieurs fois.

- Construction de carte précise : le temps de l'exploration ne sera pas pris en compte ; ceci est particulièrement vrai pour les robots utilisant des capteurs très bruités.

Nous allons parler des approches selon différents objectifs :

2.1 Minimisation du temps d'exploration

Les approches qui visent à minimiser le temps d'exploration sont des approches qui cherchent rapidement la plus grande couverture (en explorant les plus grandes zones en priorité)et celles qui cherchent la totalité de l'exploration le plus vite possible[10].

Marjovi et al.[11]ont introduit une approche pour l'exploration coopérative multi-robots, pour résoudre le problème de recherche d'incendies dans un environnement inconnu. Leur but est de cartographier l'environnement exploré tout en minimisant le temps global d'exploration, et localisant les sources d'incendie.

2.2 Minimisation de la consommation d'énergie

Les robots sont chargés par une quantité d'énergie limitée. La principale source de consommation d'énergie est l'exploration d'environnements de grande taille et l'utilisation des moteurs. Dans ce cas, l'énergie consommée est proportionnelle à la distance parcourue et au temps d'exploration. Donc, limiter la consommation d'énergie pour l'exploration d'environnement de grande taille est un aspect très important [10].

Dans l'exemple de planification éco-énergétique proposé par Mei et al[12], ont considéré deux routes R1 et R2 reliant l'emplacement A avec l'emplacement B (Figure 1).Il sont montré que malgré la route R1 est plus courte que R2, elle consomme plus d'énergie. La route R1 comporte dix segments de ligne courte, tandis que la route R2 comporte trois segments de ligne longue.

R1 a une plus courte distance avec plusieurs escales (arrêts) et détours que R2. Ceci provoque l'accélération et le ralentissement qui cause la consommation importante d'énergie. Il en résulte que malgré la route R1 est plus courte, elle consomme cependant beaucoup plus d'énergie, Cela montre la différence entre les trajets de courte distance et les chemins économes en énergie [12].

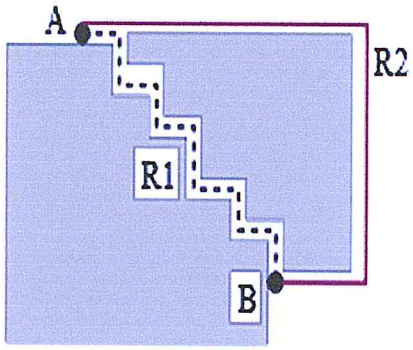


Figure 1 : Exemple de planification éco-énergétique proposé par Mei et al[12].

2.3 Maximisation la précision de la construction de cartes

L'objectif est de construire la carte la plus précise possible lors de l'exploration multi-robots. Il existe une approche qui consiste à combiner le SLAM (Simultaneous Localization And Mapping) avec une procédure d'exploration. Le SLAM, est une technique de construction de carte basée sur des estimations de pose tout en localisant simultanément les robots à l'intérieur de cette carte. La localisation active vise à guider le robot à des endroits dans le plan pour améliorer l'estimation de pose. Ceci est fait de manière répétitive à chaque observation [10].

Makarenko et al. [13] montrent que si un robot utilise une stratégie d'exploration en évitant de faire de multiples visites à la même zone, la probabilité de faire une carte la plus précise possible est réduite.

La combinaison des stratégies d'exploration et le SLAM devrait considérer de rentrer dans des espaces déjà couverts ou explorer le nouveau terrain. Ceci peut déduire qu'un système qui prend en compte cette décision, peut améliorer la qualité de la carte obtenue [10]. Cet exemple montre pourquoi l'approche qui exécute l'exploration et le retour vers des régions actives fournit de meilleurs résultats que les approches qui n'envisagent pas de rentrer dans le terrain connu pendant la phase d'exploration.

La figure 2 montre deux cartes obtenues à partir des données du monde réel acquises à Sieg Hall, Université de Washington [14]. L'image gauche représente une expérience dans laquelle le robot ne traverse la boucle qu'une seule fois avant son entrée dans le long couloir. Ce qui montre que le robot n'a pas pu correctement fermer la boucle, d'où une erreur de 7 degrés dans le sens du couloir horizontal. L'image droite montre le deuxième cas où le robot revisite la boucle ; l'erreur d'orientation a été réduite à 1 degré seulement.

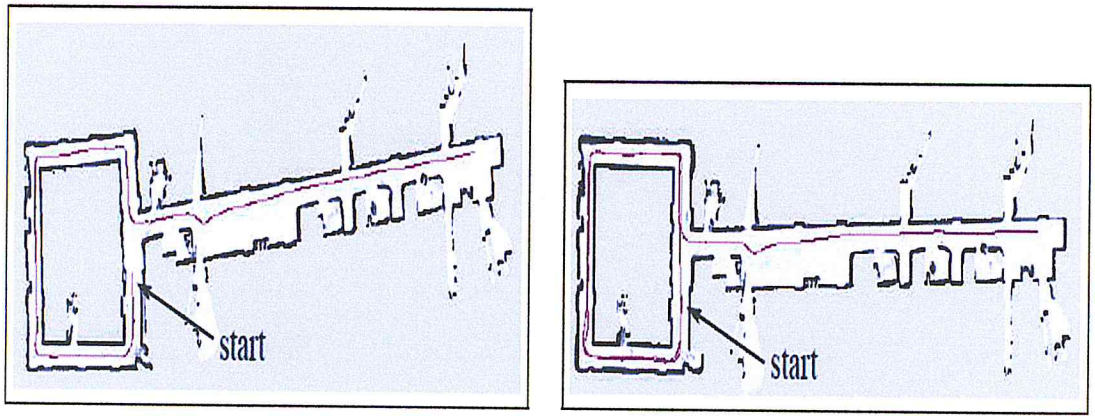


Figure 2 : Deux cartes obtenues à partir des données acquises à Sieg Hall(Washington) [14].

3. Classement des approches par hypothèses

Nous allons aborder les différentes hypothèses sur l'environnement et sur l'homogénéité et l'hétérogénéité des robots.

3.1 Types et structurations des environnements

L'environnement dans lequel se déplacent les robots a une grande influence sur la stratégie d'exploration. En fonction de l'application, le robot va agir dans des environnements cibles différents. Il en existe deux types :

3.1.1 Environnements d'intérieur

Ce sont des environnements structurés, c'est-à-dire, enrichi d'éléments spécifiques facilitant la localisation du robot, la cartographie ainsi que les chemins à suivre [15].Ce type d'environnement peut être représenté par des primitives géométriques simples (portes, murs, couloires, etc.).Parker [16] présente une solution pour effectuer une mission de surveillance et de reconnaissance à l'intérieur d'un bâtiment. Il s'agit de cartographier un étage dans un immeuble inconnu, pour rechercher des cibles ou des points d'intérêt, puis à les protéger.

3.1.2 Environnements d'extérieur

Ils ne sont pas structurés et sont considérés pour des applications en milieu vraiment naturel (urbain, polaire, forestier, etc.) [17]. Plus l'exploration s'éloigne d'un environnement structuré et statique plus elle induit des difficultés pour la mise en œuvre des robots mobiles, car il y a moins de repères contrairement à l'exploration en environnement d'intérieur [15].

Tim bailey [18] a travaillé sur la localisation et la cartographie d'un environnement immense extérieur non lisse avec l'approche de SLAM extérieur en utilisant un capteur laser à balayage 2D monté sur le pare-chocs avant d'un véhicule routier à grande vitesse où les repères observables sont exclusivement des troncs d'arbres.

3.2 Homogénéité et hétérogénéité

Les robots peuvent être homogènes ou hétérogènes. Dans le cas de robot homogènes, tous les robots sont identiques et chacun d'eux peut être remplacé par n'importe lequel. Ainsi, toutes les tâches peuvent être accomplies par n'importe quel robot du groupe. Ceci fait référence à une robustesse du système grâce à une forte redondance. Dans le cas contraire (robots hétérogènes), il faut assigner des tâches spécifiques à des robots spécifiques dotés de capacités différentes notamment au niveau physique [19]. Singh et Fujimura [20] présentent une approche décentralisée en ligne pour des robots hétérogènes. Quand un robot découvre une ouverture à une région inexplorée qu'il ne peut pas atteindre en raison de sa taille, il sélectionne un autre robot qui peut mener bien cette tâche.

4. Approches d'identification des zones à explorer

Nous décrivons ici les grandes familles d'identification des cibles à atteindre lors de l'exploration.

4.1 Approches par frontière

L'environnement peut être divisé en deux régions l'une connue, *explorée* et l'autre inconnue, *inexploré*. L'exploration consiste alors à augmenter la taille de la région explorée jusqu'à ce qu'il n'ait plus de régions inexplorées atteignables par les robots. La limite entre ces deux régions est appelée *frontières*. La coordination entre les robots est nécessaire pour que la répartition de ces derniers aux frontières soit la plus appropriée possible. L'avantage de cette approche est qu'elle vise à éviter l'assignation de plusieurs robots à des cellules frontières proches l'une de l'autre, elle est efficace et simple à mettre en œuvre. Sa limite est que parfois les robots peuvent aller vers la même frontière alors qu'il en existe d'autres qui ne sont pas affectées [10].

Yamauchi [21] est le premier à avoir développé une stratégie multi-robots d'exploration de frontières. Chaque robot se dirige vers la frontière la plus proche de lui, effectue une observation et en diffuse le résultat.

4.2 Approches par Utilité

L'approche par utilité est définie par le rapport entre l'exploration des frontières et le gain d'informations. Si un robot examine une frontière, son voisinage sera automatiquement examiné. Du coup, l'utilité des frontières est réduite si elles sont visibles depuis la frontière

désignée et s'il n'y a pas d'obstacle. Lorsque les frontières sont assez proches, un seul robot pourra faire l'exploration en une seule observation. L'avantage de cette approche est l'obtention d'une meilleure distribution des robots dans l'environnement [10].

Zlot et al. [22], Ont introduit une approche de distribution des tâches s'appuyant sur l'idée de marcher pour l'exploration multi-robots, elle introduit la notion de la minimisation du cout et la maximisation d'une utilité en même temps.

5. Approches d'affectation de cibles

Les zones à explorer peuvent être affectées de manières différentes aux robots suivant la méthode d'exploration. Les principales méthodes sont données comme suit :

5.1 Frontière la plus proche

Chaque robot se dirige vers sa frontière la plus proche pendant son exploration. Ainsi une coordination implicite s'effectue entre eux si la carte partiellement construite est partagée [10]. L'inconvénient de cette approche est que les robots peuvent s'orienter vers les mêmes frontières.

Yamauchi [21] a développé une technique dans laquelle les robots construisent une carte commune (une grille d'occupation) de manière distribuée. Son travail introduit la notion d'une frontière, qui est un emplacement à proximité d'une partie inexplorée de l'environnement. Chaque robot se dirige, ensuite, vers le centre de la région frontalière proche, mais ils le font de façon indépendante alors qu'ils partagent les cartes. À cet effet il n'y a pas de coordination explicite. Ainsi, les robots peuvent finir par couvrir la même zone et peuvent même physiquement interférer avec une autre.

5.2 Méthode Hongroise

La méthode *Hongroise* est une méthode qui simplifie la résolution du problème d'affectation c'est aussi une modélisation d'un algorithme d'optimisation combinatoire. Cet algorithme est utilisé pour l'exploration multi-robots ainsi que l'affectation de frontières aux robots, Il s'exécute sur une matrice de cout et aussi il permet de résoudre de façon optimale la minimisation de la somme des couts d'exploration de chaque robot. Sa complexité est en $O(n^3)$ où n est le nombre maximum entre le nombre de robots et le nombre de frontières[10].L'avantage de la méthode hongroise est de résoudre les conflits entre les robots c'est-à-dire lorsqu'ils se dirigent vers la même zone [2].

Wurm et al. [23] ont proposé une technique pour la coordination d'un groupe de robots qui explore l'environnement par segmentation, cette méthode prend en compte la structure de l'environnement et segmente l'environnement déjà exploré sur la base de diagramme de Voronoï. Chaque robot est assigné à un segment différent pour l'exploration de la zone correspondante. Les conflits entre les robots sont résolus par la méthode hongroise.

5. 3Méthode basé sur les échanges

C'est une approche basée sur la distribution des tâches dans un système multi-robots décentralisé. Elle est conçue pour effectuer une relation entre le robot qui demande la tâche et le robot qui est responsable de la distribution des tâches. Après une période de temps fixée, un robot analyse les demandes reçues et affecte les tâches aux robots selon un ensemble de règles.

L'avantage de cette approche est l'efficacité et la robustesse car les robots sont capables de diriger la distribution des taches grâce à un système de communication qui leur permet de communiquer entre eux, et aussi chaque robot peut prendre sa décision sans la nécessité des informations des autres robots [2].

Gerkey et Mataric[24] ont suggérer une méthode de distribution des tâches fondée sur la notion d'enchère pour une coopération multi-robots décentralisée. Le processus de coordination se déroule en cinq étapes :

- annonce d'une ou plusieurs tâches.
- évaluation métrique des différentes tâches annoncées.
- éventuel dépôt de soumission pour une ou plusieurs de tâches annoncées.
- fermeture des enchères.
- suivi d'exécution.

6. Stratégie de coordination

La stratégie de coordination se réfère à l'équilibre des robots qui sont expédiés à chaque objectif d'exploration [25]. L'objectif dans la coordination de l'exploration de plusieurs robots est de maximiser le gain d'informations attendu au fil du temps.

L'objectif d'un processus d'exploration est de couvrir l'ensemble de l'environnement dans un minimum de temps. Par conséquent, il est essentiel que les robots gardent trace de quelles zone de l'environnement ont déjà été explorées. En outre, les robots doivent construire une carte afin de planifier leur parcours et de coordonner leurs actions [26].

6.1 Modes de communications entre les robots

La communication est un moyen de coordination dans les environnements multi-robots. Elle peut se faire selon différentes architectures. Il est possible de les classifier selon deux types :

6.1.1 Architectures centralisées

Dans l'architecture centralisée, un agent central est responsable des informations globales concernant l'environnement ainsi que les robots (Figure 3). Il construit la carte partielle, calcule les couts pour chaque robot et communique avec chacun d'eux pour leur permettre de partager des informations [27]. L'avantage de cette architecture centralisée est de fournir une vue globale du monde dans lequel les robots évoluent mais elle est efficace pour un petit nombre de robots [2].

Wurm et al. [28] utilisent une méthode ou chaque robots fournis des informations qui vont être ensuite récolté par une unité centrale cette dernière construit la carte partielle, calcule les couts et distribue des taches à chaque robots. Ainsi, durant l'exploration la communication est centralisée vers le seul et unique robot leader, ce rôle peut être attribué à différents robots.

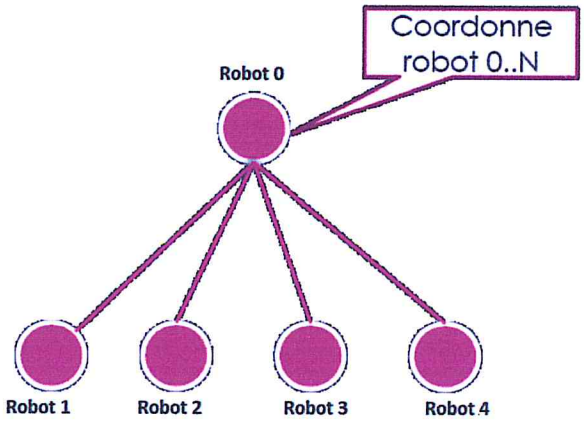


Figure3 : Architecture centralisée [27]

6.1.2 Architecture décentralisée

Il existe deux types dans les architectures décentralisées : les architectures distribuées et les architectures hiérarchiques.

a) Architecture distribuée

Dans les architectures distribuées, il n'existe aucun agent de contrôle central la responsabilité de contrôle est distribuée, tous les robots sont égaux en matière de contrôle et tous sont individuellement autonomes dans leur processus de prise de décision (Figure 4). Cette architecture permet de facilité, élargir le nombre de robots et avoir la capacité de la tolérance de panne[29].

Silvia et al. [30] ont proposé un système multi-robots M+ qui utilise une architecture distribuée ou chaque robot possède sa propre connaissance du monde et peut décider de ses actions futures en tenant compte de son contexte actuel et de la tâche à accomplir.

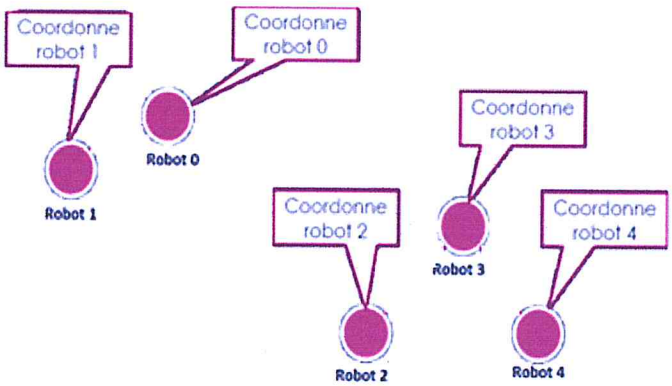


Figure4 : Architecture distribuée [27]

b) Architecture hiérarchique

C'est une architecture hybride, entre les deux architectures centralisée et distribuée. Dans cette architecture il existe un ou plusieurs agents de contrôle centraux. Ces derniers appartiennent à des sous parties de l'espace de travail appelées clusters. Un seul agent de contrôle est responsable de l'organisation de chaque cluster.

Cette architecture permet de maximiser la coordination de centralisation de l'architecture distribuée [17].

Fukuda et al.[31] ont proposé un système multi-robot CEBOT qui utilise une architecture hybride dans laquelle les robots sont physiquement couplés avec d'autres robots pour former des clusters. Dans ces clusters, des robots leaders sont sélectionnés afin de coordonner l'exécution des tâches.

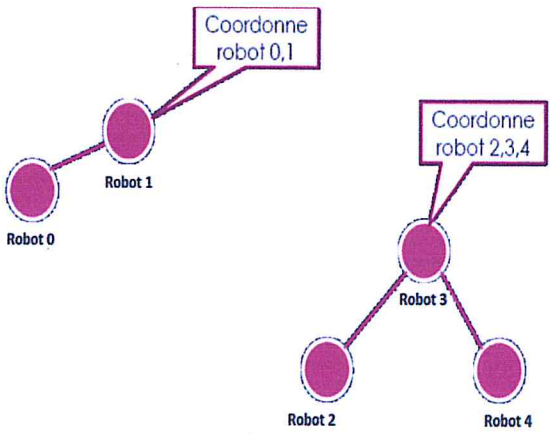


Figure 5 : Architectures hybrides [27].

6.2 Type de communication

La communication est une réaction réciproque entre les robots. Ainsi, les robots peuvent partager les informations de leurs positions ainsi que l'environnement avec les autres ; et aussi chaque robot peut obtenir des informations des autres robots sur leurs objectifs et leurs actions [2]. La communication entre les robots peut être réalisée implicitement ou explicitement :

6 .2.1 Communication explicite

Elle se réfère à la nature pour la réalisation des échanges directs d'informations entre les robots, c'est-à-dire de robot à robot sans passer par un tiers[2]. Madhavan et al.[32] ont décrit un algorithme basé sur le filtre de Kalman étendu (EKF) pour la localisation d'une équipe de robots à travers un réseau local sans fil. Gerkey et Mataric [33] ont présenté une structure générale pour les équipes de robots mobiles coopératifs pour la communication explicite inter-robots et l'attribution dynamique de tâches

6.2.2 Communication implicite

Elle se réfère au mode selon lequel un robot reçoit des informations des autres robots durant l'exploration au travers de l'environnement. Ce mode est effectué en incluant multiples types de capteurs. Il se divise, ainsi, en deux catégories [2] :

a) Communication implicite active

Elle désigne les cas dans lesquels les robots communiquent en recueillant des informations laissées par d'autres robots dans l'environnement.

b) Communication implicite passive

Elle désigne les cas dans lesquels la perception des changements de l'environnement correspond à une communication entre les robots. Chaque robot calcule les informations des autres robots (la position et l'attitude) à travers des données perçues.

Hollinger et al.[34] ont présenté un algorithme d'approximation en utilisant une coordination implicite, cherchant à résoudre le problème de la planification de trajectoire multi-robots dans des environnements intérieurs connus.

6.3 Différents rôles que jouent les robots

Les robots peuvent jouer deux rôles différents dans un système multi-robots :

6.3.1 Leader

Il est supposé avoir une trajectoire prédéfinie ; il est suivi par tous les autres robots et est souvent plus capable qu'eux. Lorsque le leader change de comportement, les autres robots auront besoin d'ajuster leurs comportements en conséquence, il donne ainsi des commandes à travers un canal de communication [27].

6.3.2 Follower

Le rôle du Follower est de suivre les activités ou les actions du robot Leader en l'observant et en même temps en travaillant pour atteindre son objectif [27].

Koo et Shahruz [35] ont effectué l'approche leader-follower et ont suggéré une méthode pour piloter un groupe de véhicules aériens sans pilote (UAV) dans une formation souhaitée. Le leader et qualifié par rapport aux autres il peut calculer le chemin de chaque (UAV) lui seul a des capteurs et des caméras il communique aux autres UAV à travers un canal de communication, le cheminement qu'ils doivent suivre.

7. Cartographie collaborative par un groupe de robots mobiles

Les robots utilisent l'ensemble de leurs capteurs afin de modéliser leurs environnements dans lequel ils doivent se déplacer et agir ; ceci est indispensable pour la localisation ainsi la cartographie. La cartographie est l'étape permettant la construction d'une carte qui reflète l'architecture de l'environnement à partir des différentes informations recueillies par le robot lors de son exploration. Dans cette section, nous allons décrire les différentes représentations possibles d'une carte.

7. 1 Types de représentation

Les trois principaux types de représentation sont donnés dans ce qui suit :

7.1.1 Carte métrique

Le but de l'approche métrique est de produire une carte géométrique plus ou moins détaillée de l'environnement à partir des données perçues. Elle contient des informations généralement définies dans un référentiel unique (longueur, distance, position, etc.)[36]. La carte métrique représente l'environnement par un ensemble d'objets de n'importe quelle taille ou n'importe quelle forme auxquels sont associées des positions dans un espace métrique, généralement en deux dimensions. Ces cartes offrent une relation bien définis au monde réel. Les objets mémorisés dans la carte peuvent être très divers (obstacles que le robot a rencontré, etc.) [15]. L'avantage des cartes métrique est la représentation de l'ensemble de l'environnement. La totalité de cette représentation permet donc d'estimer avec clarté et de manière continue la position du robot sur l'ensemble de son environnement. Leur inconvénient est qu'elles sont difficiles à construire et à mettre à jour [17].

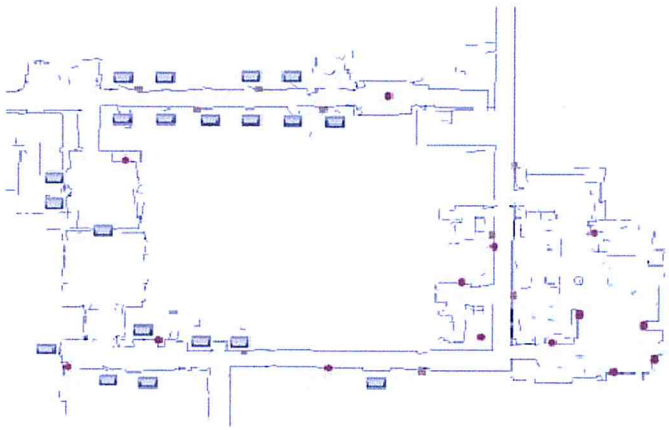


Figure 6 : Exemple de carte métrique du monde [39].

Castellanos et al. [37] ont construit une carte métrique et extrait des points, des coins, des segments et des surfaces à partir des informations sensorielles (laser, vision, etc.). Janet et al. [38] ont construit une carte métrique en utilisant les réseaux de neurones afin de rassembler et regrouper des données obtenues par le sonar.

Il existe deux types de la carte métrique :

a) Corrélation de Scan

La première méthode simple de cartographie métrique consiste à utiliser une méthode de corrélation de cartes locales. Lors de l'exploration du robot dans une zone qui n'est pas

encore cartographiée, la corrélation s'effectue entre la carte locale courante et la carte locale précédente. La nouvelle carte locale est, ensuite, ajoutée à la carte de l'environnement à sa position estimée. Dans le cas où le robot revient à une zone déjà cartographiée, la corrélation est faite entre la nouvelle carte locale et la partie de la carte globale la plus proche de la position actuelle [39]. Cette représentation est généralement suffisante lorsqu'elle est exploitée avec les données d'un télémètre laser et aussi dans un espace limitée qui ne contient pas de grands chemins. Son inconvénient est lorsque l'exploration est terminée, des erreurs de localisation apparaissent ce qui ne permet pas trouver la bonne portion de carte avec laquelle faire la corrélation [29].

Figure7 : Exemple de carte métrique créée par corrélation de scans [39].

b) Grille d'occupation

Les grilles d'occupation constituent une représentation surfacique simple et populaire. Cette représentation permet la décomposition de l'environnement en un ensemble de cellules [36]. Chaque cellule contient un indice (probabilité, histogramme, etc.) indiquant si l'espace correspondant est plutôt libre ou occupé et aussi l'emplacement des obstacles [17]. La grille d'occupation représente un modèle qui est apte de faire la mise à jour de l'environnement à une fréquence élevée, et permettant de réviser facilement les probabilités d'occupation, donc de suivre l'évolution de l'environnement autour du robot, ce qui est indispensable pour une meilleure réactivité [36]. L'avantage des grilles d'occupations est qu'elles sont utilisables pour différentes formes de représentation des environnements, et elles produisent des informations d'occupation sur la localisation des obstacles. Leurs mises à jour se font de manière facile et rapide car elles sont économiques en calcul. Elles sont généralement faciles à interpréter par l'homme. L'inconvénient des grilles d'occupation est l'absence de densité

c'est-à-dire elles sont adaptable pour la représentation d'environnements encombrés et inutile dans les espaces immenses vides [17].Elfes [40] a proposé cette méthode de cartographie en une représentation en deux dimensions de l'espace exploitable pour un robot mobile en utilisant des capteurs à ultrasons. Par la suite, Elfes a proposé un développement appelé« la grille d'inférence », où il a affecté à chaque cellule un vecteur contenant plusieurs informations (sa probabilité d'occupation, sa classe d'appartenance, etc.). Birk et al. [41] ont proposé de construire une seule grille globale depuis plusieurs grilles locales déjà faites par plusieurs robots. Stepan et al. [42] ont fait la fusion entre sonar, laser et sonar, caméra dans un environnement d'intérieur pour construire une grille d'occupation solide pour une meilleure planification.

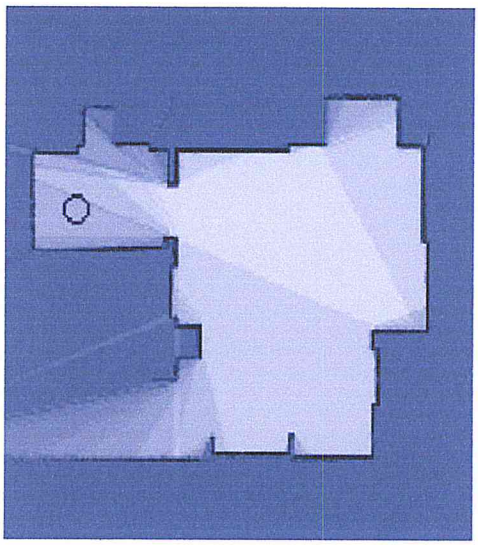


Figure8 : Exemple de grille d'occupation [39].

7.1.2 Carte topologique

L'approche topologique est une représentation plus abstraite interprétant les relations entre les éléments de l'environnement, sans l'utilisation d'un référentiel absolu [43].La carte topologique est une représentation de l'environnement sous forme de graphe. D'où ses nœuds correspondent à des positions que le robot peut atteindre ainsi que les arrêtes reliant les nœuds marquent la possibilité pour le robot de passer directement d'un lieu à un autre [17]. Les environnements structurés sont facile à modéliser d'une façon topologique, car ils conduisent à une structure topologique avec moins d'ambiguïtés (le passage d'une pièce à un couloir)[36]. L'avantage des cartes topologiques est l'existence de la certitude sur le mouvement des robots c'est-à-dire les incertitudes ne s'accumulent pas généralement étant donné que le robot exécute une exploration locale, entre endroits. Les cartes topologique

présente éventuellement une partition des éléments de l'environnement ce qui les rend facile à comprendre par l'homme, par contre leurs inconvénient est le manque d'informations spatiale ce qui va empêcher le robot d'effectuer des raisonnements géométriques sur tout l'environnement c'est-à-dire le robot peut rencontrer des difficultés en sélectionnant la trajectoire optimale entre deux endroits [17].

Kuipers et al. [44] ont utilisé une boussole et 16 sonars pour différencier les différents nœuds. Par contre chez Kortenkamp et al. [45], les sonars ne sont pas suffisants pour effectuer l'identification de nœuds. Pour cela, ils ont suggéré une caméra perspective, et un petit nombre d'images pour chaque situation sera suffisant pour garantir une bonne identification.

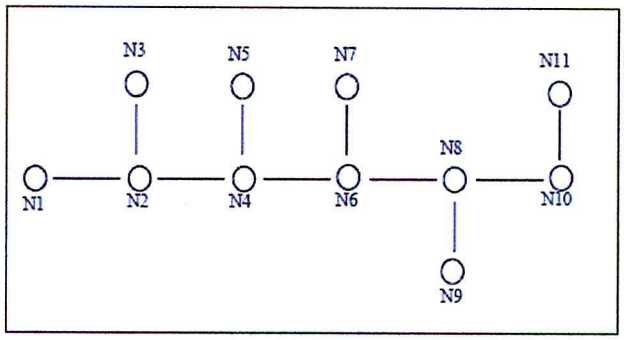


Figure9 : Exemple carte topologique [36].

7.1.3 Représentation hybride

La représentation hybride est un mélange entre les deux catégories précédentes, topologie et métrique. Les représentations topo-métrique peuvent être considérée comme hydrique, c'est à dire des cartes topologiques, contenant des informations métrique sur les arrêtes du graphe. Les types de représentations sont souvent hiérarchiques : sur la base d'une carte métrique, une carte topologique est construite ; puis, les nœuds sont classifiés suivant leurs types et permettent de mémoriser les objets associés.

L'intérêt de ces représentations est de contenir des cartes métriques précises pour des zones plus simples à cartographier [29]. Ces cartes présentent l'avantage de pouvoir utiliser deux ou plusieurs types de représentation à la fois, ce qui permet d'avoir des avantages de chacune de représentation, et aussi de s'épargner des limites [46].

L'exemple suivant (figure 10) montre une carte hybride qui mélange les cartes métriques locales pour les pièces et la carte topologique globale. Les positions relatives des cartes locales ne sont pas précises, car les couloirs ne sont pas cartographiés précisément, mais simplement suivis pour aller d'une porte à une autre [29].

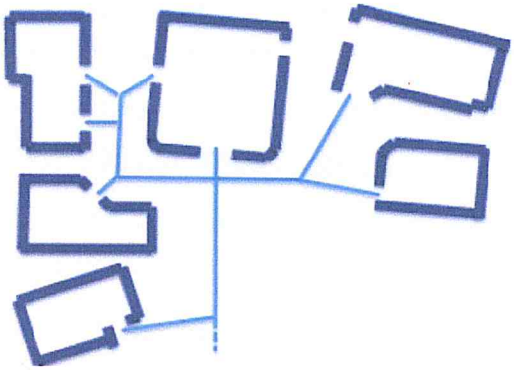


Figure10 :Exemple de carte hybride [29].

7.2 Construction de carte globale

En multi-robots la cartographie d'un environnement quelconque est un des principaux problèmes, il s'agit donc de l'intégration des données collectées par les différents robots opérant dans différentes parties de l'environnement. L'intégration des informations peut se produire à différents niveaux ceci consiste à la fusion des informations de haut niveau à partir de données extraites de bas niveau brutes recueillies par des capteurs [47]. Pour une construction collaborative de carte par un groupe de robots il est nécessaire d'établir une fusion de carte [48].

G. Dedeoglu et G. Sukhatme [49] ont décrit un algorithme de point de repère pour la correspondance de cartes qui consiste à combiner des cartes topologiques de l'environnement intérieur.

H. Hajjdiab et R. Laganiere [50] ont utilisé une approche basée sur la vision afin de combiner les cartes créées par une équipe de robots dans le même chantier.

7.2.1 Carte fusion

C'est une tâche qui consiste à construire un modèle cohérent d'un environnement en associant des données connues entre les caractéristiques de carte locales construite par différents robot. Si les emplacements initiaux des robots sont connus, ou acquis au cours de rendez-vous, la carte fusion est une extension assez simple d'une cartographie de robot unique. Si les robots ne connaissent pas leurs positions relatives, l'accent est mis sur l'extraction des caractéristiques à partir des données brutes du capteur. Car on ne sait pas comment et où les traces des robots doivent être connecté [9][51].

échouent à se rencontrer, l'hypothèse d'emplacements relatifs est rejetée et la fusion de carte n'est pas exécutée.

c) *Carte Fusion lorsque les positions relatives des robots ne sont pas connues*

Les deux approches qu'on a décrites ci-dessus exigent que les positions relatives des robots soient connues au début ou les positions sont acquises au cours de la cartographie. Sans l'information de position les approches ne peuvent pas fusionner les cartes. La tâche de la fusion de cartes sans aucune information de position est un cas plus difficile qui nécessite des solutions différentes, la plupart des méthodes de fusion de cartes de ce type utilisent l'approche des transformations. Ils tournent et traduisent une des cartes contre l'autre pour trouver la meilleure correspondance possible.

Les cartes qui sont le plus souvent utilisées pour la cartographie sont cartes métriques et cartes topologiques décrites précédemment. Dans la pratique, la fusion des cartes topologiques est avéré être plus facile que la fusion des cartes métriques [9].

Dedeoglu et Sukhatme [49] sont l'un des premiers chercheurs à résoudre le problème de la fusion des cartes topologique. Ils ont développé l'algorithme qui correspond aux paires de sommets pour trouver la meilleure transformation possible des cartes. La meilleure transformation est avec le plus grand nombre de sommets qui se chevauchent. Seuls les sommets d'un type similaire sont comparés, d'où l'espace de transformation est considérablement réduit.

Une autre approche est conçue par Ho et Newman [48]. En plus des informations géométriques, les robots recueillent aussi des informations sur l'apparition visuelle de l'environnement. Les images sont recueillies et liées selon les parties des cartes métriques. Quand il est nécessaire de fusionner les cartes locales de deux robots, les images sont comparées. Afin d'éviter de fausse correspondance, les informations métriques sont prises en compte. Cependant, il n'est pas pratique d'utiliser cette approche si les informations visuelles de l'environnement ne sont pas distinctives.

Xin Ma et al.[60]ont utilisé une approche pour le problème de la fusion de deux cartes partielles sans cadres de références communes et des informations de position relative des robots, ils ont utilisé un algorithme génétique adaptatif pour trouver la zone de chevauchement entre les grilles d'occupations partielles pour réaliser la fusion de cartes.

8. Conclusion

Tout au long de ce chapitre, nous avons abordé de différents objectifs et les travaux les plus réputé dans le domaine de l'exploration multi-robots : la couverture maximale et l'exploration complète en un temps minimum. Nous avons aussi parlé de différentes architectures : centralisée, décentralisé etc. Ensuite nous avons mis l'accent sur l'état de l'art des différentes méthodes de représentation des environnements : carte métrique, carte topologique, etc. Nos travaux de recherche se placent donc dans le thème de l'exploration et la cartographie d'un environnement par un groupe de robots mobile. Nous avons constaté que les stratégies centralisées permettent une exploration efficace car il existe un seul agent central qui a l'information complète sur le système, cette stratégie est simple et facile à implémenter. Nous avons découvert que les cartes métriques ont une représentation meilleure car elles permettent de représenté la géométrie du monde de manière explicite.

Dans le chapitre suivant nous allons présenter le simulateur utilisé Player/Stage afin d'accomplir notre travail de l'exploration et de la cartographie multi-robot. Nous allons introduire une étude conceptuelle qui dispose de différents diagrammes UML, ainsi qu'une implémentation qui se base sur une méthode de détection d'obstacles et de cartographie.

Chapitre 2

Conception et Implémentation

Chapitre 2
Conception et implémentation

1. Introduction

L'application initiale visée par ce travail consiste en l'exploration et la cartographie d'un environnement d'intérieur inconnu par un groupe de robots mobiles. Nous allons proposer une stratégie de déploiement et d'exploration visant à minimiser le temps d'exploration total (temps nécessaire à la construction de la totalité de la carte) et maximisant la précision de la carte globale construite. Pour cela, les robots doivent se répartir au mieux dans leur environnement afin d'explorer simultanément différentes zones de cet environnement.

Dans le chapitre précédent, plusieurs types de cartographie et d'architecture sont été présentés. Parmi les types de cartes discutés, nous avons sélectionné la carte métrique ; et parmi les différentes architectures, nous avons choisi l'architecture centralisée. Ces deux types représentent le modèle le plus adéquat pour réaliser notre objectif de la cartographie.

Ce chapitre est composé de deux parties. La première est dédiée à la conception de la solution proposée ; la deuxième partie est consacrée à l'implémentation de cette solution.

2. Choix du modèle et d'approche de cartographie

La cartographie robotique consiste en l'acquisition des modèles spatiaux des environnements physiques à travers les robots mobiles. Pour construire une carte, les robots doivent être équipés de capteurs qui leurs permettent de percevoir le monde qui les entoure.

Afin d'accomplir la tâche de la cartographie, les données collectées par les différents robots situés dans des zones différentes de l'environnement doivent être intégrés. L'intégration des données consiste à fusionner les informations qui sont recueillies par les capteurs [47].

2.1 Modèle métrique

Nous avons choisi le modèle métrique car il est le mieux adapté aux environnements d'intérieurs et aux environnements structurés. C'est un modèle qui contient des informations métriques sur l'environnement.

L'utilisation des cartes métriques permet de représenter la géométrie du monde de manière explicite, c'est-à-dire des représentations exactes ou approximatives. Elles fournissent une représentation détaillée de l'environnement à partir des données perçues [36].

Ces cartes contiennent des informations métriques comme les endroits estimés, ou la longueur des chemins entre eux. Une carte métrique peut être une représentation incluant tout type d'objets de n'importe quelle taille ou n'importe quelle forme [15].

Ces cartes sont souvent plus faciles à interpréter par l'homme car elles offrent une relation bien définie avec le monde réel. Leurs utilisations présentent toutefois l'avantage que les informations recueillies ont une sémantique plus forte. En effet, ces informations caractérisent la structure spatiale de l'environnement [17].

2.2 Approche centralisée

Nous avons opté pour une approche centralisée. Celle-ci est la représentation la plus fidèle du monde qu'il est possible d'obtenir. Cette représentation est une mise en commun de l'information reçue de chacun des robots. Ainsi, elle est plus robuste que l'approche distribuée puisqu'il est impossible, par exemple, que deux robots se voient attribuer un même rôle ou encore une même zone de couverture. Cette approche peut être avantageuse et très efficace pour les petits groupes de robots, elle permet de fournir une vue globale du monde dans lequel les robots évoluent [2].

La caractéristique principale de cette approche est que toutes les données sont envoyées par les robots à un seul serveur central. Celui-ci décide de la stratégie à adopter et émet les signaux de commande destinés à tous les robots. Sachant qu'il doit aussi s'occuper de récupérer toutes les données fournies par les différents robots, il est essentiel que ce serveur soit informé en permanence de l'état et la position des robots acquise via un capteur laser, c'est-à-dire la distance et l'angle par rapport aux murs et aux obstacles. La prise en compte de toutes ces données facilite ainsi la cartographie. Ce serveur a une vue globale donc la solution proposée peut être l'optimale.

3. Serveur Player

Il existe un serveur de réseau pour le contrôle des robots nommé Player. Celui-ci fournit une interface claire et simple pour les capteurs et les actionneurs du robot [69].

Il permet de faciliter la programmation en comparaison avec la programmation robotique. Dans cette dernière, le programmeur doit être responsable d'écrire un module pour récupérer les données des capteurs, un module pour traiter ces données et un module pour générer les commandes de contrôle des moteurs. Par contre, dans le modèle de la programmation par Player, on doit écrire seulement le module pour traiter les données obtenues par Player et lui donner des commandes. Le module pour récupérer les données des capteurs et le module pour générer les commandes de moteurs sont réalisés par Player (voir Annexe) [27].

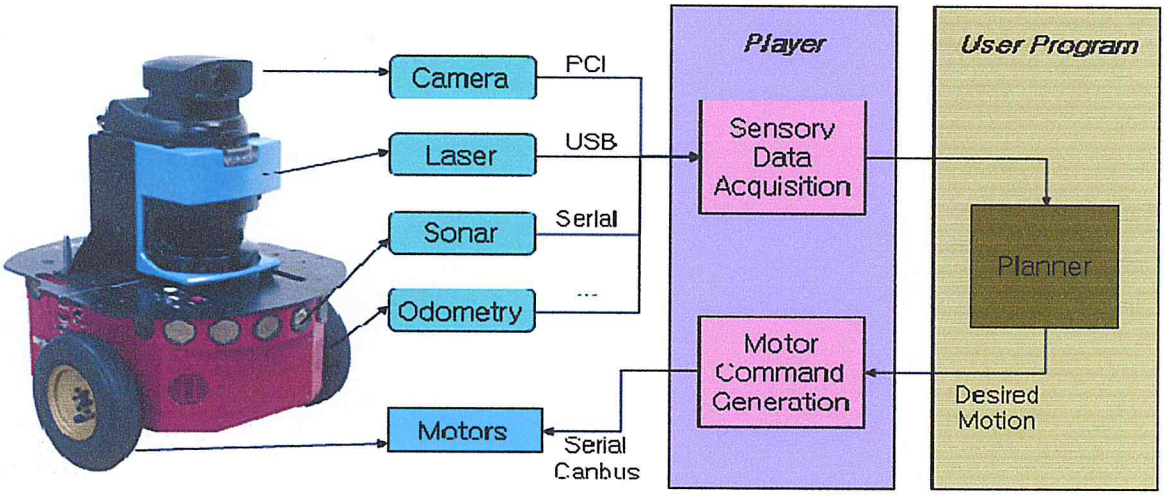


Figure 1 : Modèle de la programmation par Player [27]

Player fonctionne sous le modèle de Client/serveur comme montré par la figure 2. Ce modèle a plusieurs avantages :

- Le modèle Client/serveur du Player permet aux programmes qui contrôlent le robot d'être écrit dans n'importe quel langage de programmation et de fonctionner sur n'importe quel ordinateur avec une connexion réseau au robot.
- Player fournit également des mécanismes de transport qui permettent l'échange de données entre les pilotes et les programmes de contrôle qui s'exécutent sur des machines différentes. Le transport le plus couramment utilisé est maintenant un transport basé sur socket TCP client/serveur [62].

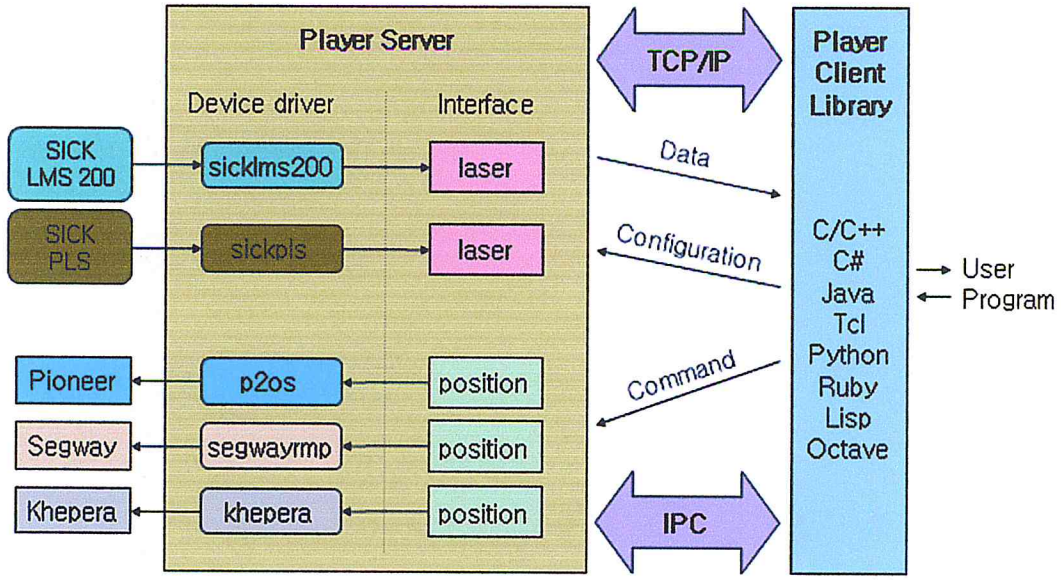


Figure 2 : Modèle client/serveur [63]

4. Étude conceptuelle

L'étude conceptuelle se définit généralement comme une étude de premier niveau et d'évaluation préliminaire d'un projet. Elle a pour but de développer les caractéristiques essentielles de la solution à émerger.

Afin de modéliser notre solution, nous avons choisi le formalisme UML (Unified Modeling Language). Celui-ci est un langage de modélisation qui permet de construire plusieurs modèles d'un système indépendamment des techniques d'implémentation mise en œuvre par la suite. Certains montrent sa structure interne, d'autres montrent le système du point de vue des utilisateurs, d'autres aussi offrent une vision détaillée ou globale. Les modèles sont complémentaires et peuvent être rassemblés. Ils sont élaborés tout au long du cycle de vie du développement d'un système.

UML dispose de neuf diagrammes qui permettent de modéliser les aspects structurels, les aspects dynamiques, et les aspects propres à la représentation des exigences fonctionnelles.

Son ensemble de diagrammes très formels est l'un des avantages qui lui permet de représenter l'architecture et le fonctionnement des systèmes informatiques complexes en tenant compte des relations entre les concepts utilisés et l'implémentation qui en découle[64].

Nous avons choisi de faire la modélisation avec trois diagrammes : diagramme de cas d'utilisation, diagramme de séquence et diagramme de classes.

4.1. Diagramme de cas d'utilisation

Il décrit la façon dont le système sera utilisé. C'est aussi un modèle qui permet une meilleure représentation des interactions entre les acteurs du système et le système lui-même [64].

Nous avons choisi trois représentations de diagramme de cas d'utilisation afin de représenter les interactions entre les acteurs et le système. Elles sont données comme suit :

4.1.1. Connexion et affectation des robots aux zones

L'application vise à cartographier un environnement par un groupe de robots mobiles, pour cela le client doit d'abord se connecté aux robots qui sont actif après avoir fait leurs configurations (adresse IP, PORT et INDEX), tous les robots utilisent le même Port. Pour cela chaque robot doit avoir un index pour pouvoir le différencier. Une fois les robots connectés, Player va les affecter aux zones différentes afin d'équilibrer l'environnement et pouvoir l'explorer entièrement.

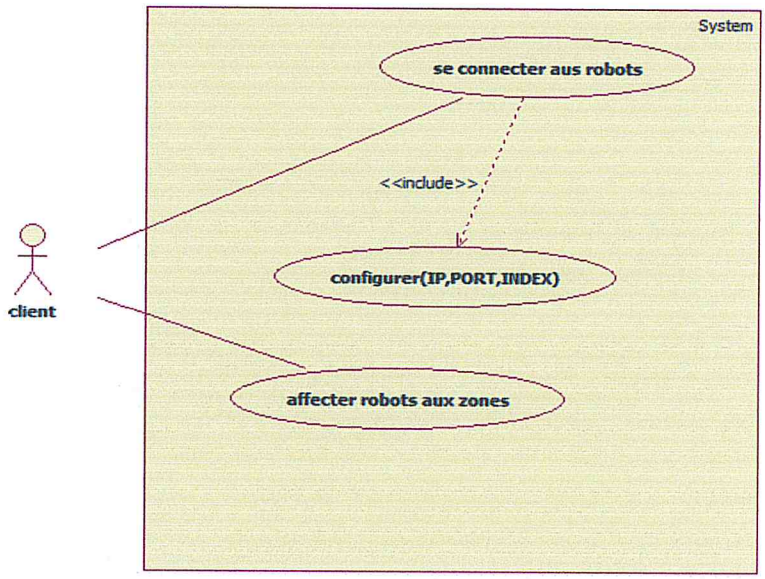


Figure 3 : Connexion et affectation des robots aux zones.

4.1.2. Récupération des données

Après avoir établie la connexion et positionné les robots dans des zones différentes, l'exploration commence. Chaque robot explore une zone et en même temps le serveur Player va récupérer leurs données (position des robots, position des murs) et les fournir par la suite au client.

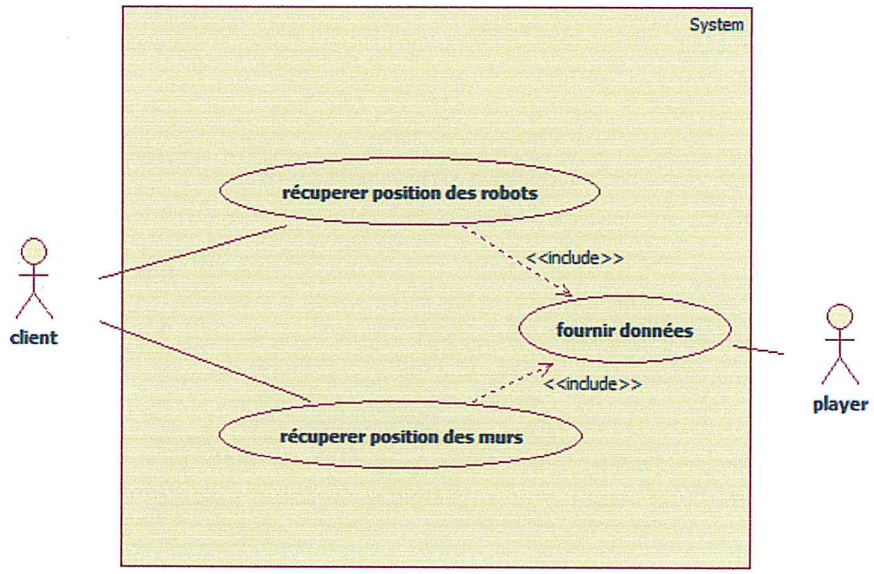


Figure 4 : Récupération des données.

4.1.3. Construction de la carte

La carte globale de l'environnement sera construite au fur et à mesure que les données récoltées par les robots (position des murs et des obstacles) soient récupérées et traitées par le client.

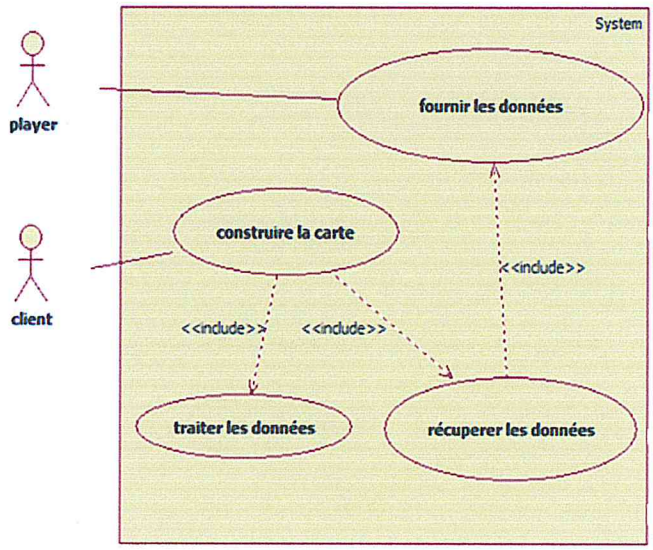


Figure 5 : Construction de la carte.

4.2. Diagramme de séquence

Il décrit une interaction entre plusieurs objets organisés dans le temps. Cette interaction est un ensemble de messages échangés entre les objets pour effectuer une

opération ou obtenir un résultat. Il sert à illustrer un cas d'utilisation et aussi il est l'une des vue la plus importante d'UML [64].

Nous avons choisi deux représentations de diagramme de séquence afin de montrer les interactions entre les objets. Ces deux représentations sont données comme suit :

4.2.1. Connexion et commande des robots

Avant toute chose le robot doit être connecté au Player. Une fois la connexion est établie, le client fera une configuration pour qu'il se connecte au Player.

Player sert d'interface pour les dispositifs de robot. Pour cela, le client souscrit les dispositifs et par la suite envoie une commande au robot qui est l'affectation à une zone.

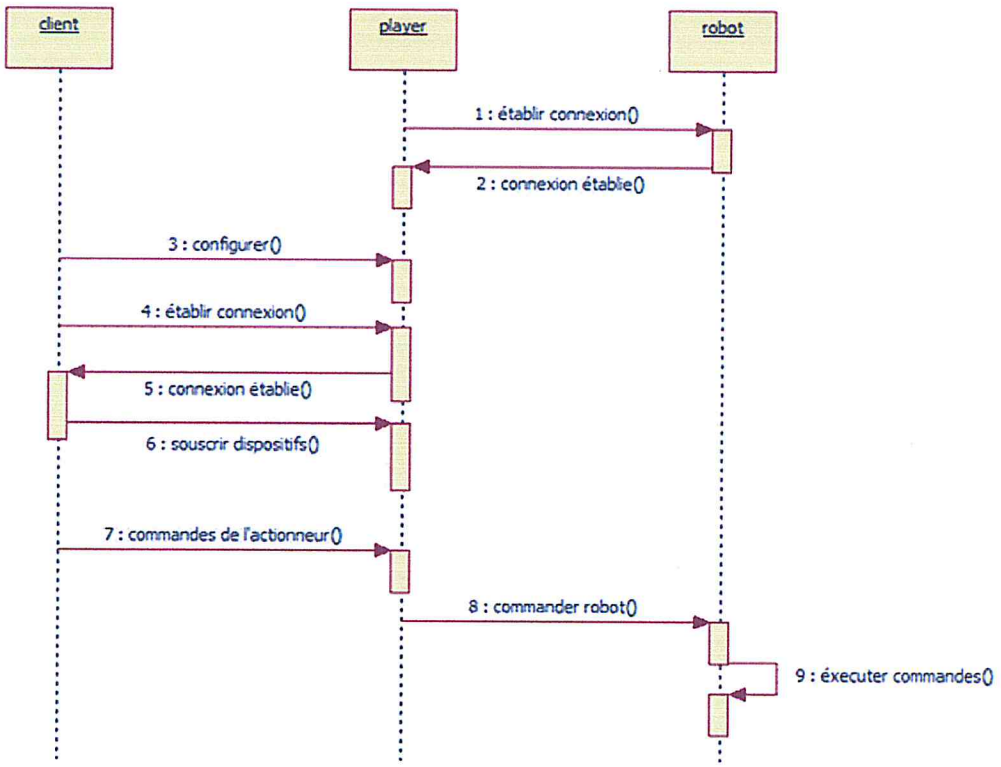


Figure 6 : Connexion et commande des robots

4.2.2. Récupération des données et construction de la carte

Lorsque le client souscrit les dispositifs et récupère par la suite les données, il les traite pour en construire la carte de l'environnement.

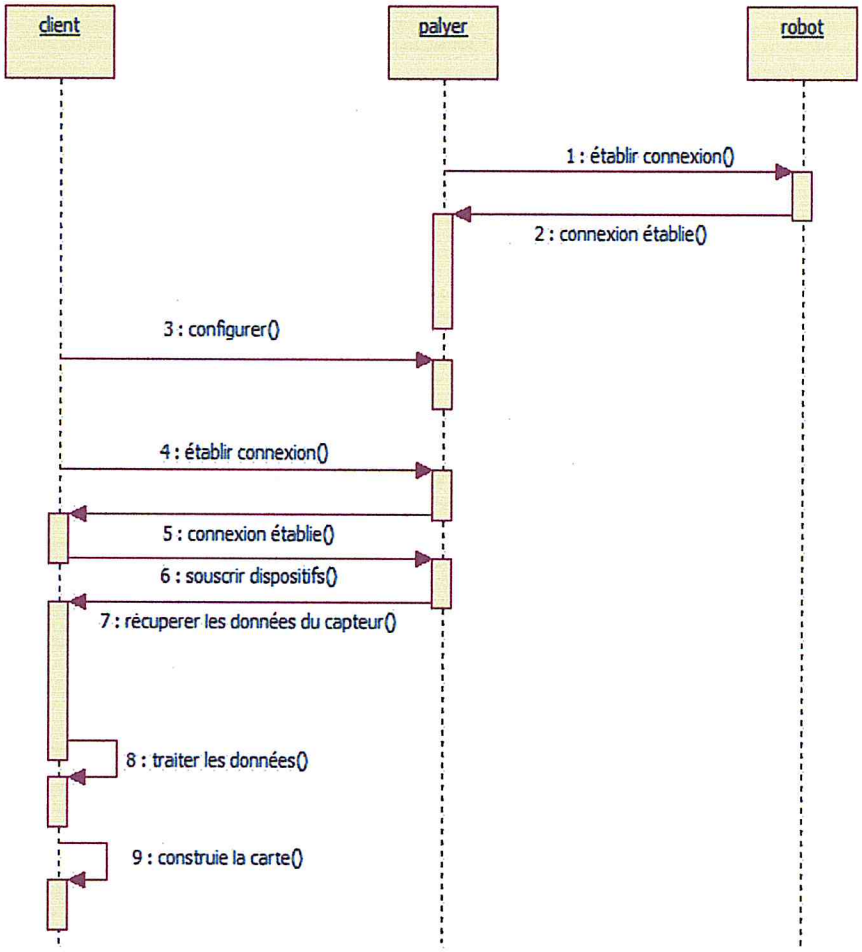


Figure 7 : Récupération des données et construction de la carte

4.3. Diagramme de classe

Il est considéré comme le plus important de la modélisation orientée objet. Il permet de modéliser de façon statique une collection d'éléments qui montre la structure du modèle en termes de classes et de relations entre ces classes. Le diagramme de classe est présenté comme suit :

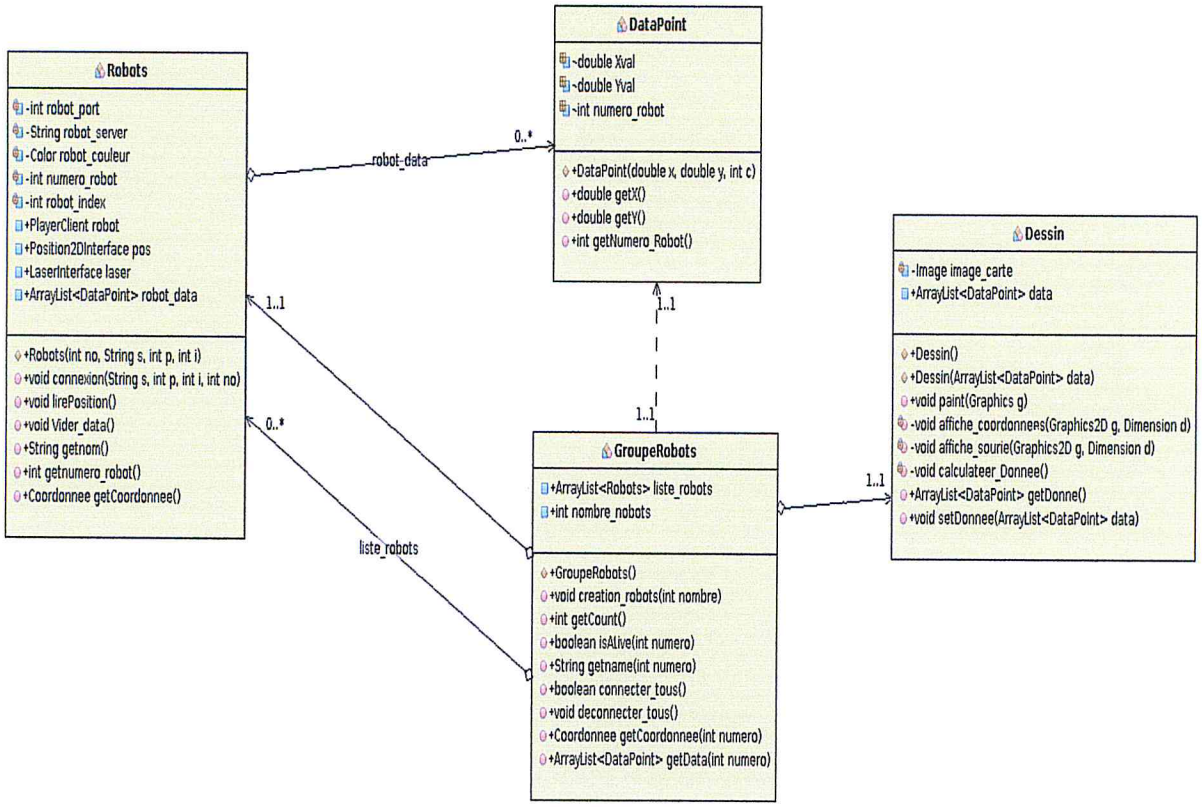


Figure 8 : Diagramme de classes.

5. Implémentation

En ayant comme objectif de faire une construction collaborative d'une carte par un groupe de robots mobiles. Nous avons choisi de cartographier le rez-de-chaussée de la Division Productique et Robotique au niveau du Centre de Développement des Technologies Avancées (CDTA). Nous avons travaillé en simulation à cause de manque des matériels. Afin de simulé les robots et l'environnement désiré, nous avons utilisé *stage* qui permet la simulation de l'environnement en deux dimensions conçu pour s'interfacer avec le serveur Player décrit précédemment.

5.1. Simulateur Stage

Stage peut être utilisé pour simuler une variété de robots, une variété de capteurs dans une variété d'environnements. Il prend en charge la définition simple des nouveaux environnements de test et les variables de matériel de robot. Le paquet de stage est conçu pour se coupler avec l'interface du Player afin de permettre à l'utilisateur de simuler des robots et des comportements de robots sans avoir besoin de matériel physique. Les modèles de robots

en Stage sont conçus de telle sorte qu'ils répètent les attributs physiques importants, tels que la taille du robot, et la forme (voir Annexe).

Stage a été spécifiquement conçu pour soutenir la recherche dans les systèmes multi-robots. Lors de la programmation et l'expérimentation avec de nombreux robots, des avantages du développement rapide sont multipliés, Stage permet des expériences avec de grandes populations de robots qui seraient trop coûteux à acheter et à entretenir [65].

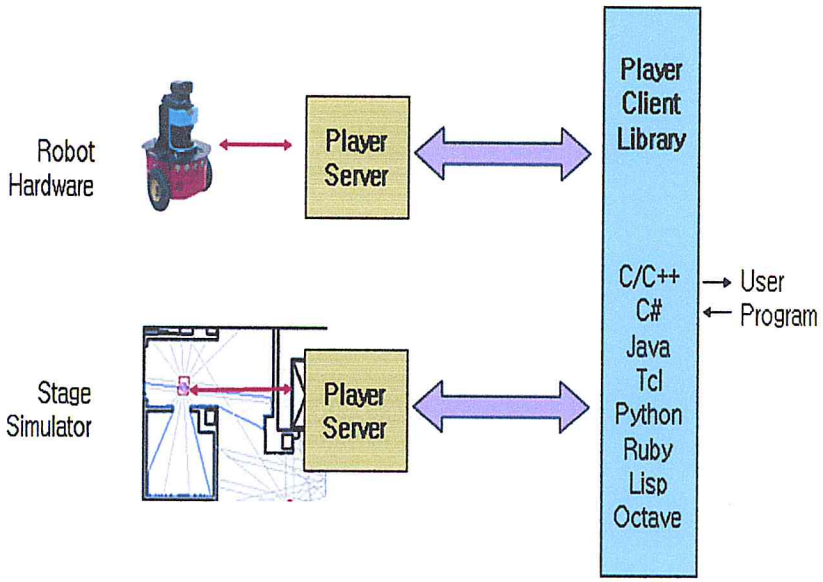


Figure 9 : Abstraction matérielle [63].

5.2. Fichier de configuration

Player/Stage comprend trois types de fichiers nécessaires pour la réalisation de l'environnement simulé et la description des robots :

- Fichier.world
- Fichier.cfg (fichier de configuration).
- Fichier.inc(fichierinclude).

5.2.1. Fichier .world

Dans le fichier.world nous définissions toutes les informations nécessaires sur l'environnement que la description des robots.

a) **Définition de l'environnement** : la définition de notre environnement est donnée comme suit :

```
# load an environment bitmap
floorplan
(
  name "RDV"  ->1
  size [32.000 22.000 0.800] ->2
  pose [0 0 0 0] ->3
  bitmap "CDTA.png" ->4
)
```

1. Représente le nom de l'environnement simulé
2. La taille choisie de l'environnement
3. La position de l'environnement dans le plan
4. L'image correspondante à l'environnement simulé

Figure 10 : Définition de l'environnement.

b) Description d'un robot : la description du robot simulé est donnée par la figure suivante :

```
pioneer2dx
(
  # can refer to the robot by this name
  name "r0" ->1
  color "red"  ->2
  pose [ -10.540 6.590 0 64.065 ]  ->3

)
```

1. Nom du robot simulé
2. Couleur du robot simulé
3. Position initiale du robot simulé

Figure 11 : Description du robot simulé.

5.2.2 Fichier.cfg

Il représente ce que le Player lit pour obtenir toute les informations sur le ou les robots que nous avons utilisé (nom, les interfaces utilisées). Il indique aussi au Player quels drivers doit il utiliser pour interagir avec le robot. La figure 12 ci-dessous illustre la description de notre fichier.cfg que nous avons créé.

```
driver
(
  name "stage"  ->1
  provides [ "simulation:0" ]
  plugin "stageplugin"

  # load the named file into the simulator
  worldfile "simple2.world"    ->2
)

# Create a Stage driver and attach position2d and laser interfaces
# to the model "r0"
driver
(
  name "stage"
  provides [ "position2d:0" "laser:0" "graphics2d:0"  ]  ->3
  model "r0"   ->4
  alwayson 1   ->5
)
```

1. Nom du driver
2. Fichier world utilisé
3. Les interfaces utilisées
4. Modèle du robot simulé
5. Indicateur de simulation

Figure 12 : Fichier de configuration.

5.2.3Fichier.Inc

Il suit la même syntaxe et le format d'un fichier. World, mais il peut être inclus dans ce dernier, qui est facilement réutilisable. Dans notre cas nous avons inclue trois fichier.inc dans le fichier.world comme suit :

```
include "pioneer.inc" ->1
include "map.inc" ->2
include "sick.inc" ->3
```

1. La description totale du robot
2. La définition du plan
3. Description totale du capteur

Figure 13 : Fichiers inclues dans le fichier.world.

5.3. Robots et capteur simulés utilisés

Les robots choisis sont de type Pioneer 2DX (voir figure 14).Le robot réel a les caractéristiques suivantes :

- poids 20 kg.
- vitesse maximale est de 1.6m/s.
- deux roues différentielles.

La fiabilité et la durabilité de ce robot en fait la plateforme de référence pour la recherche en robotique [66].

Figure 14 : Robot Pioneer 2DX [66]

Il est équipé d'un télémètre laser. Ceci est le moyen le plus répandu en robotique mobile pour obtenir des mesures précises de distance. Le télémètre laser est composé d'un émetteur et d'un récepteur, et il évalue le déphasage entre l'émission et la réception [67].

Son principe de fonctionnement est qu'à un instant donné, une impulsion lumineuse très courte est envoyée. La réflexion de cette onde donne un écho qui est détecté au bout d'un temps proportionnel à la distance capteur-obstacle. La direction des impulsions est modifiée par rotation d'un miroir, l'angle de balayage de 180 degrés et la portée du capteur est de 8 mètres.

5.4. Interfaces et programmation

Pour la programmation, nous avons utilisé NetBeans. C'est est un outil de développement puissant, qui se distingue par sa facilité d'utilisation due à son ergonomie qui utilise toutes les technologies possibles d'aide au développement. NetBeans est un environnement complet placé en open source, incluant toutes les fonctionnalités et toutes les technologies liées à Java permettant un développement rapide et visuel des applications java. Il a été lié avec Player en ajoutant les bibliothèques correspondantes. Les deux interfaces du Player utilisées sont les suivantes :

- *Position2dProxy* pour obtenir la position dans laquelle se trouve le robot (coordonnées X et Y, et l'angleθ) ainsi que pour contrôler les mouvements du robot.
- *InterfaceLaser* pour obtenir des informations du dispositif laser, qui mesure les distances séparant les robots des obstacles.

5.5. Position d'obstacles

Pour obtenir la position d'un obstacle, nous avons utilisé les ratios sinus et cosinus, ainsi que la distance (rayon), et les angles $\phi_j$et θ_r. Il est nécessaire d'ajouter la position du robot (xr, yr) pour l'obtention de la position exacte. Comme le montre les équations suivantes :

$$xj = x_r + rayon_j * \cos(\theta_r + \phi_j) ;$$

$$yj = y_r + rayon_j * \sin(\theta_r + \phi_j) ;$$

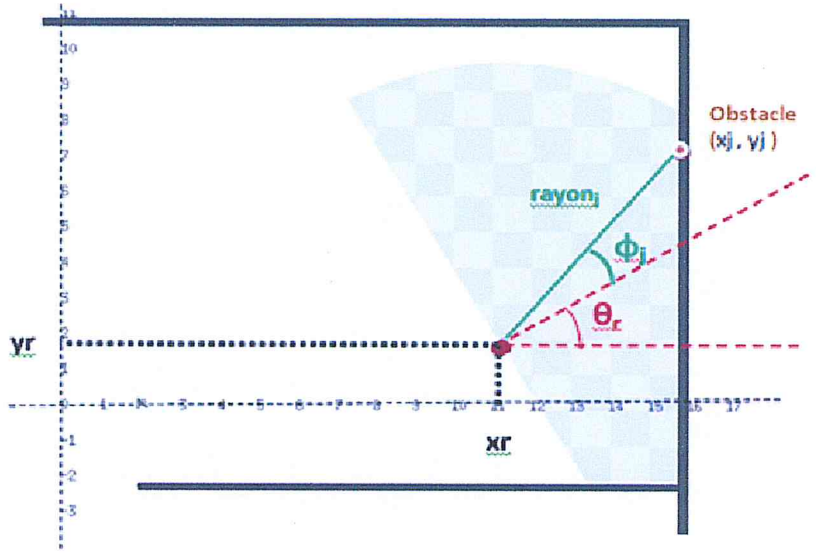


Figure 15 : Détection d'obstacles.

6. Algorithme de cartographie

6.1. Définition des variables

DataPoint(x,y, n) //(x,y)sont les coordonnées du point dans la carte

//n représente le n° du robot qui a détecté ce point

Robot

r_i : Playerclient //interface de connexion avec Player server

pos_i : Poisition2dinterface //interface de connexion avec le positionneur du robot

$laser_i$: InterfaceLaser //interface de connexion avec le capteur laser du robot

Serveur : chaîne de caractère //détermine l'adresse IP du robot par défaut localhost

Port : entier //le port de connexion par défaut 6665

Index : entier //détermine l'index du robot dans l'environnement Stage

$Liste\_data_i$: liste<Datapoint>//la liste des coordonnée des obstacles détectés par le robot

X_r, y_r, θ_r : réels //position et orientation du robot r

FOV : réel //champ de vision du robot (180°)

φ_j : réel //angle de décalage du rayon laser par rapport l'orientation du robot

Nmax : entier //nombre de rayons dans le faisceau laser (360 rayons)

RayonMax : réel //la portée du laser (8m)

xj, yj : réel //position des obstacles détectés

rayon : liste<réel> //distance entre les obstacles et le robot

GroupeRobot : liste <Robot>//ensemble des robots connectés dans le stage

Carte : liste<Datapoint> //ensembles des obstacles détectés par tous les robots

6.2. Algorithme de construction cartographique

L'algorithme de cartographie que nous avons développé est donné comme suit :

1. **Initialiser les paramètres nécessaires à la connexion**

   (Nombre de robot, serveur, port, index, FOV, portée)

2. **Création des instances pour tout le groupe de robots**

   Pour i = 1 à nombre_robotsFaire

   Créer r_i ;

   Créer pos_i ;

   Créer $laser_i$;

   Ajouter r_i au GroupRobot ;

   FinPour

3. **Connecter tous les robots**

   Pour i = 1 à nombre_robotsFaire

   Connecter r_i ;

   FinPour

4. Si (r_in'est pas actif) aller (3)

5. **Pour tous les robots lire la position, les données du laser et détecter les obstacles**

   Pour i = 1 à nombre_robots Faire

   Lire x_r, y_r, θ_r ;

   Lire rayon ;

   Pour j = 1 à NmaxFaire

   Si ($rayon_j$<RayonMax) Alors

   // Calcul des coordonnées de l'obstacle

   $\Delta\varphi$ = FOV / Nmax ;

   φ_j= FOV/2 – j * $\Delta\varphi$;

   xj= x_r + $rayon_j$ * cos (θ_r+ φ_j) ;

   yj= y_r + $rayon_j$ * sin (θ_r+ φ_j) ;

   Ajouter Datapoint (xj, yj, i) à $Liste\_data_i$;

   FinSi

   FinPour

   Ajouter tous les points de $Liste\_data_i$ à la carte ;

   FinPour

6. Si (ce n'est pas la fin de la simulation) aller à (5)
7. Fin.

Remarques

- *Playerclient*, *Poisition2dinterface* et *InterfaceLaser* sont des classes importées du package javaclient3.
- Pour de différents robots physiques, c'est l'adresse du serveur qui fait la différence entre les robots.

7. Développement

L'environnement de simulation Player/Stage a été utilisé pour mettre en œuvre l'algorithme de cartographie en raison de sa robustesse et de la clarté de l'exécution. Au début, il est nécessaire de créer la simulation du Stage (comme nous l'avons cité précédemment).Dans notre cas, nous avons créé le rez-de-chaussée de la DPR du CDTA, qui sera utilisé pour la navigation des robots. La simulation est exécutée à partir d'un terminal linux (commande "playersimpl.cfg") comme montré par la figure 16.

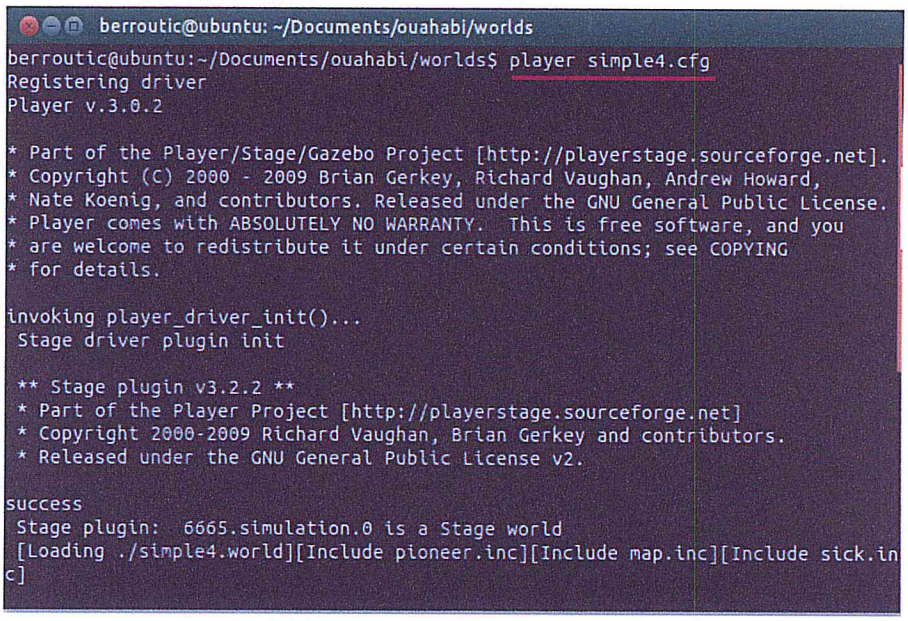


Figure 16 : Terminal linux.

L'environnement est composé par sept salles et un couloir partagé. Il est exploré par quatre robots comme le montre la figure ci-dessous :

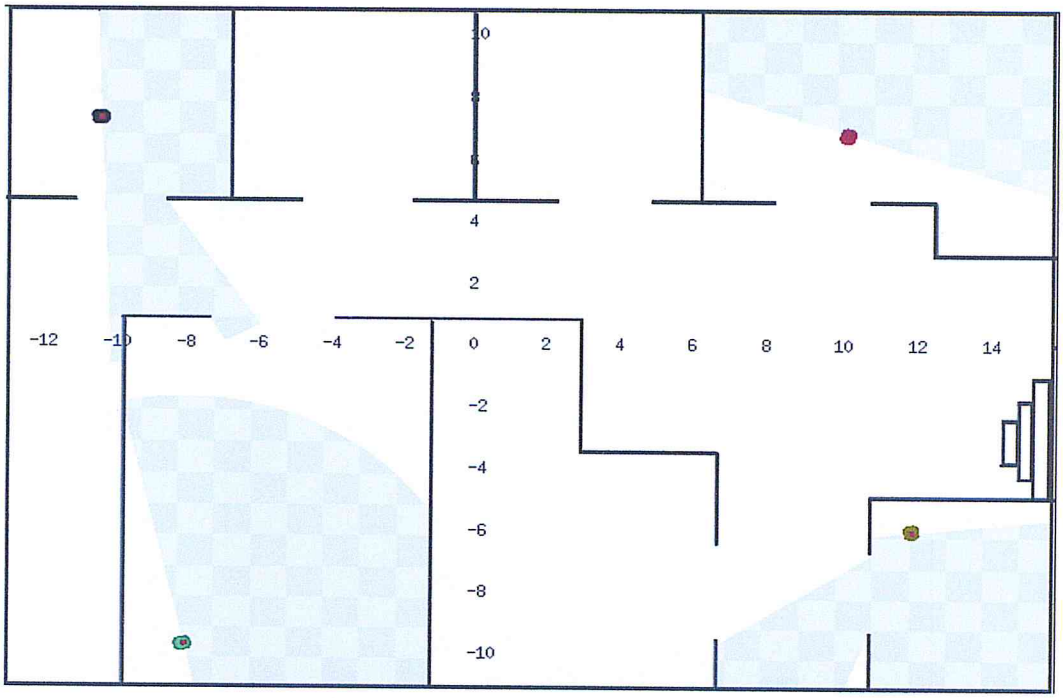


Figure 17 : Environnement simulé.

Une fois démarré la simulation, il est nécessaire de lancer l'application, celle-ci affichera une interface comme suit (Figure 18) :

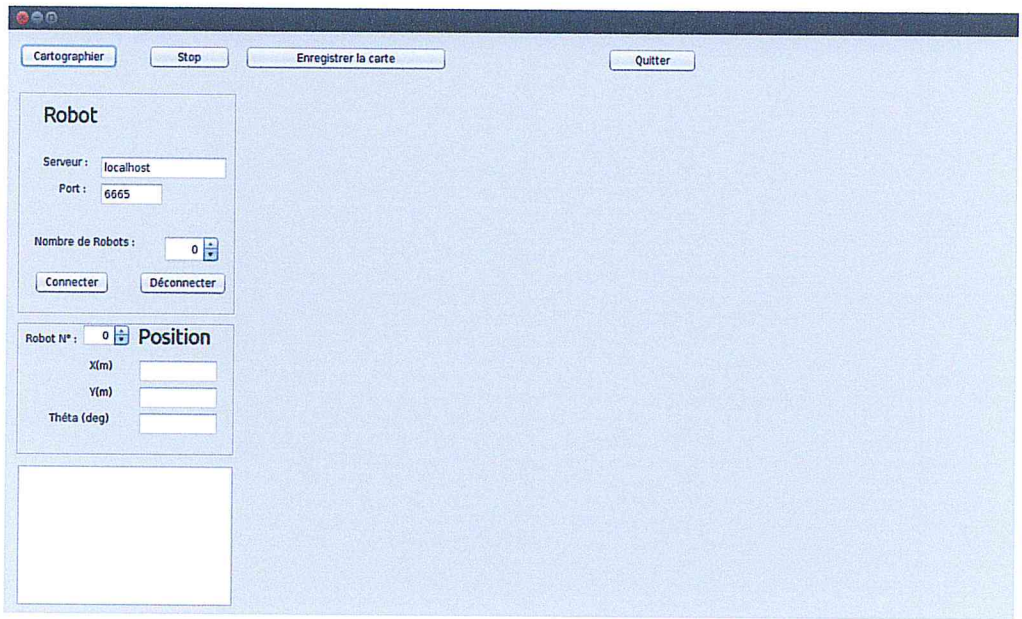


Figure 18 : Interface de l'application.

Avant tout, il est nécessaire de choisir le nombre de robots, puis se connecter avec au serveur Player. Une fois la connexion établie, on peut alors commencer la cartographie. La figure suivante illustre l'environnement cartographié par les robots.

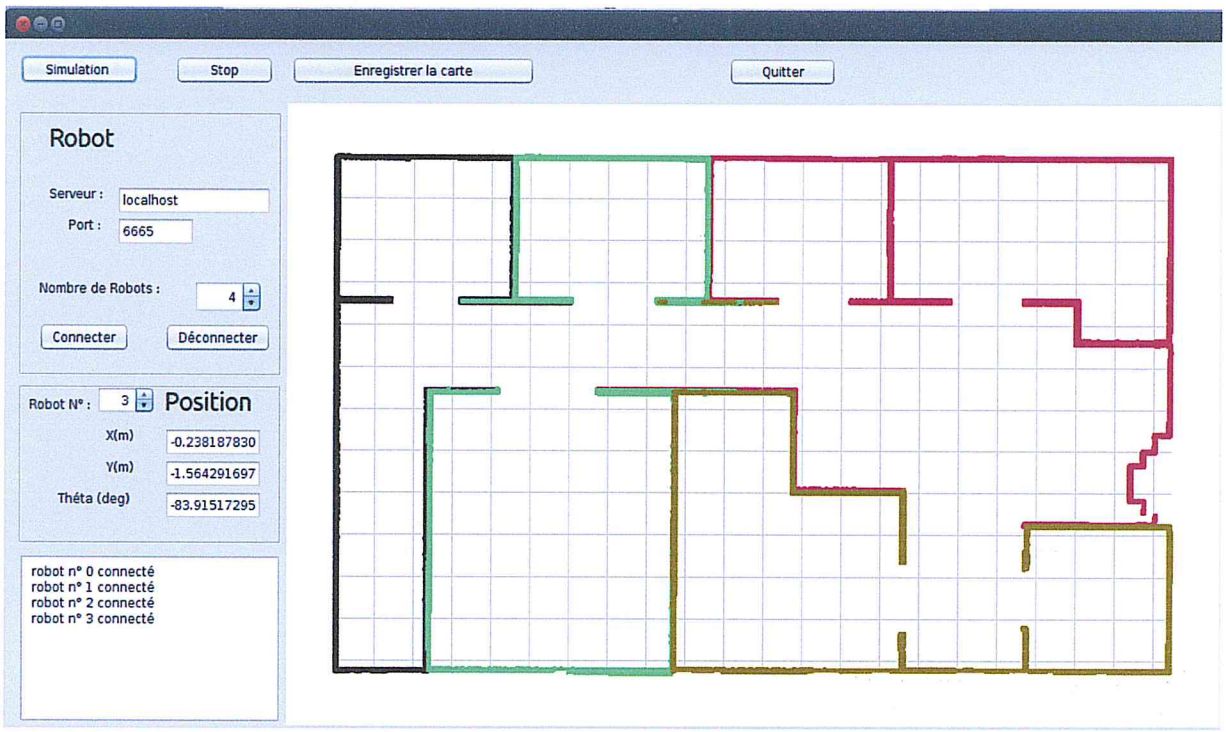


Figure 19 : Environnement cartographié.

Nous remarquons que chaque robot a cartographié la zone qu'il a explorée ;par exemple ; la partie en vert a été cartographiée par le robot vert de la figure 17, etc.

8. Conclusion

Dans ce chapitre, nous avons présenté notre travail concernant l'exploration et la cartographie multi-robots. Les réflexions de ce travail se basent sur une architecture centralisée et une représentation métrique de carte.

Dans la première partie, nous avons présenté l'étude conceptuelle qui se focalise sur la modélisation par le langage UML de la solution proposée. Nous avons présenté la méthode de récupération des données et la cartographie ainsi que toute l'application par des différents diagrammes.

Dans la deuxième partie, nous avons présenté le simulateur utilisé Player/Stage ainsi que les robots simulés et les interfaces du Player utilisées. Nous avons, ensuite, présenté une formule pour la détection des objetsde l'environnement qui se base sur le calcul des distances

fournies par les capteurs Laser équipant les robots. Après, nous avons expliqué l'algorithme général utilisé pour la construction de cartes de l'environnement (cartographie). Enfin, nous avons présenté l'interface utilisateur de l'application que nous avons développée.

Dans le chapitre suivant nous allons évaluer en simulation les performances de l'algorithme utilisé par le biais de plusieurs scénarios afin de calculer le temps nécessaire et la précision de la carte globale générée.

Chapitre 3

Tests et Validation

Chapitre 3
Tests et validation

1. Introduction

L'expérimentation a pour but d'établir un comparatif entre l'utilisation d'un système mono-robot et un système multi-robots lors de la construction de la carte globale d'un environnement quelconque.

Dans ce chapitre nous allons présenter les expérimentations que nous avons menées pour valider notre travail présenté dans le chapitre précédent. Toutes les expériences rapportées dans ce document ont été réalisées avec le simulateur Player/Stage (voir annexe).À cet effet, deux types d'environnements ont été utilisés (*i*) environnement structuré simple et (*ii*) environnement structuré complexe avec plusieurs scénarios.

Dans chaque scénario, nous avons changé le nombre de robots et nous avons utilisé en les déployant de deux manières différentes (*i*) déploiement aléatoire ou(*ii*) déploiement manuel. Le test se termine ainsi lorsque les robots construisent la totalité de la carte.

Les différents résultats obtenus sont comparés en termes de temps nécessaire pour la construction de la carte globale ainsi que sa précision. Le temps de construction est donné comme suit :

$$Temps = Temps\ de\ fin\ de\ construction - Temps\ de\ début\ de\ construction$$

La précision de la carte globale construite est calculée comme suit :

$$Précision = (1 - taux\ d'erreurs) * 100$$

Tel que :

$$Taux\ d'erreurs = \frac{\sum taille\ des\ erreurs\ de\ la\ carte}{\sum tailles\ des\ segments\ de\ la\ carte + \sum taille\ des\ erreurs\ de\ carte}$$

- erreur : représente un obstacle non réel.
- Segments : représente tous les obstacles de l'environnement réel.

Il y a lieu de préciser que tout au long de chapitre, les positions des robots mobiles sont données en mètre.

2. Validation dans un environnement structuré simple

Ces simulations consistent en l'exploration et la cartographie d'un environnement inconnu d'intérieur et ce en utilisant un groupe de robots de type Pioneer 2-DX. Chaque robot est équipé d'un télémètre laser fournissant 360 échantillons dans un champ de vue de 180 degrés avec une portée maximale de 8 mètres.

2.1 Premier cas : Utilisation d'un seul robot mobile

Dans le premier cas, nous avons utilisé un seul robot mobile qui explore la totalité de l'environnement.

2.1.1 Premier scénario : Déploiement aléatoire du robot mobile

Dans ce scénario, la position est générée d'une manière aléatoire (X=-4.9 et Y=3.6).La figure suivante montre le robot entrain d'explorer l'environnement.

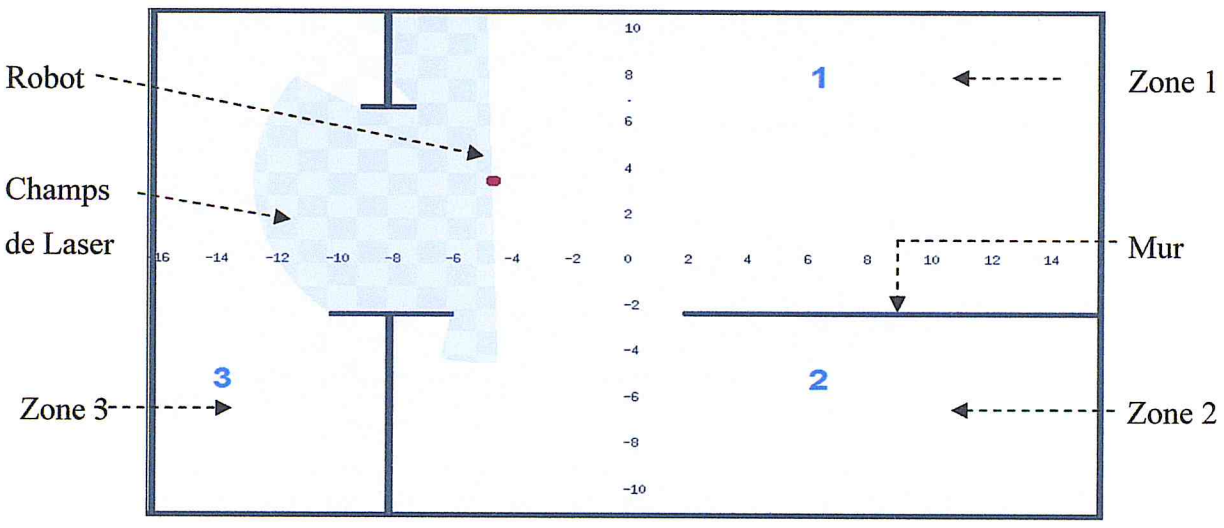


Figure 1 : Environnement exploré par un robot

La construction de la totalité de la carte par le robot a durée 496 secondes et sa précision est de 100% ; elle est donnée par la figure 2 ci-dessous :

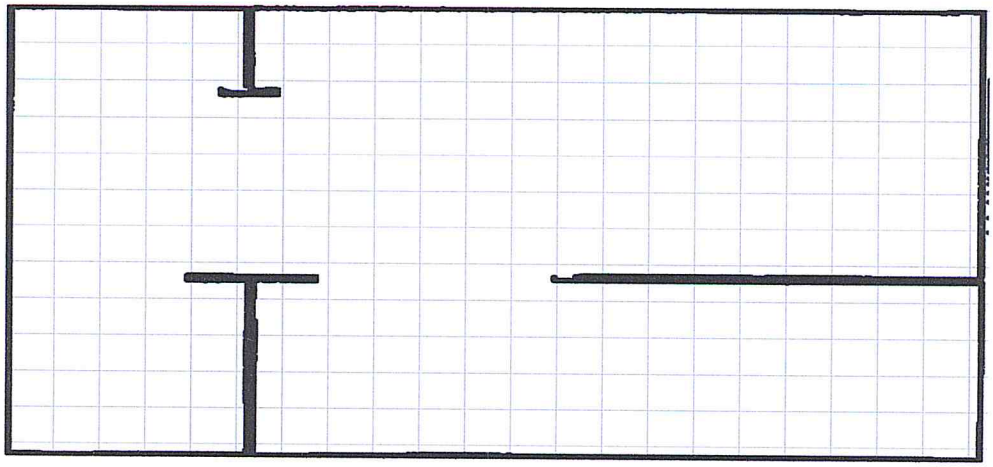


Figure 2 : Carte construite par le robot (temps de construction = 496s)

2.1.2 Deuxième scénario : Déploiement manuel du robot mobile

Par contre dans le deuxième scénario, la position du robot est assignée à une zone d'une façon manuelle. Initialement, nous avons affecté le robot mobile à la zone 1 ; ses coordonnées initiales sont (X=13.5 et Y=0.5). La durée de la construction de la totalité de la carte est de 540 secondes et sa précision est de 100% ; elle est donnée par la figure 3 suivante :

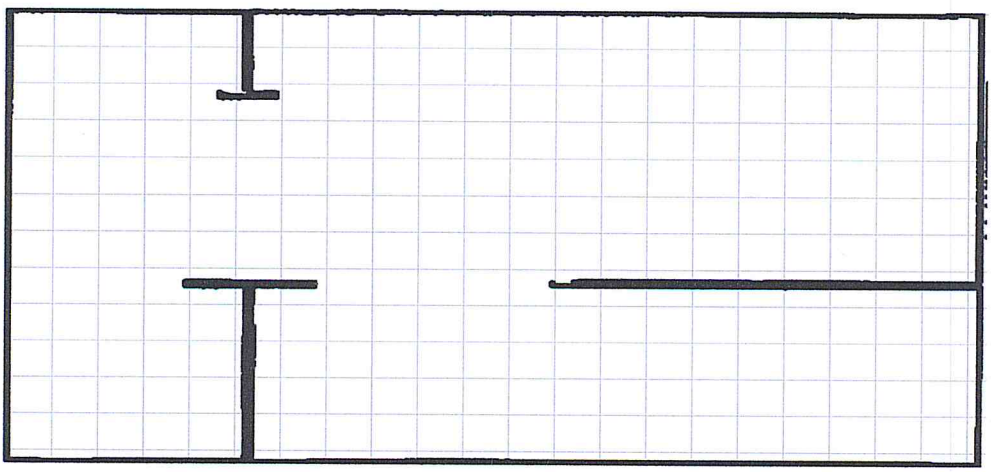


Figure3 :Carte construite par un robot (temps de construction = 540s)

2.2 Deuxième cas : Utilisation de deux robots mobiles

Ce deuxième cas consiste à utiliser deux robots mobiles, qui explorent et cartographient l'environnement en parallèle.

2.2.1 Premier scénario : Déploiement aléatoire des deux robots mobiles

Les positions initiales des deux robots sont générées d'une manière aléatoire. Les positions du premier et du deuxième robot obtenues sont respectivement (X=-2 et Y=2.1),(X=-11.9 et Y= 3.1). La figure 4 suivante montre les deux robots entrain d'explorer l'environnement.

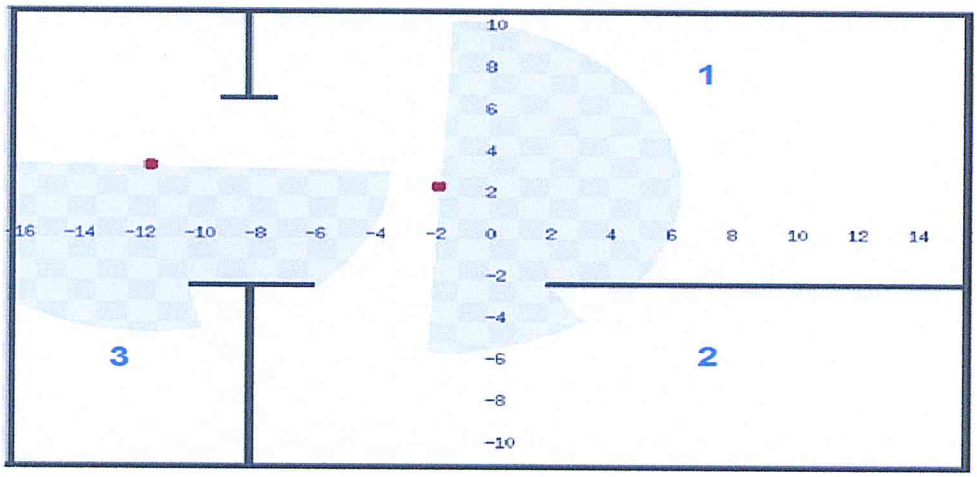


Figure 4 : Environnement exploré par deux robots

La construction de la carte globale par les deux robots a durée 324 secondes et sa précision est 94% ; elle est donnée par la figure suivante :

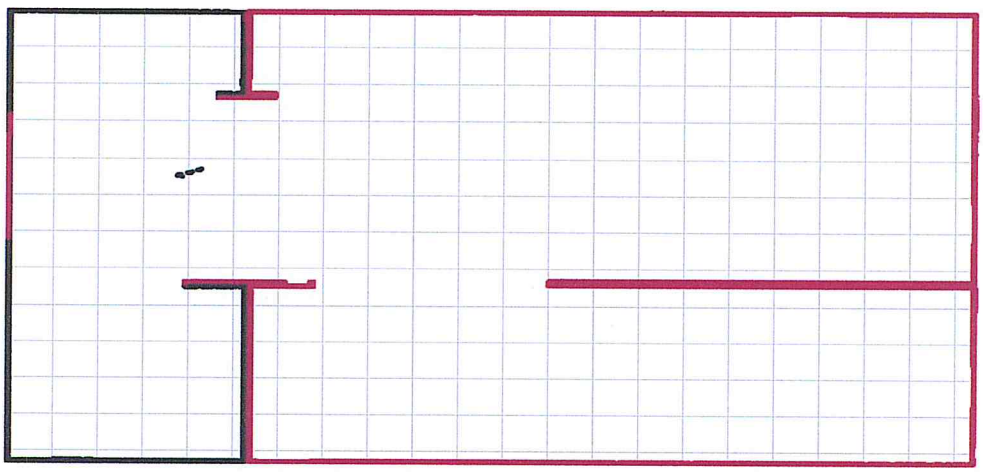


Figure 5 : Carte construite par les deux robots (temps de construction = 324s)

La couleur rouge consiste en la zone explorée par le premier robot ; la couleur noir est affecté à la zone explorée par le deuxième robot.

2.2.2. Deuxième scénario : Déploiement manuel des deux robots mobiles

Nous avons utilisé le même nombre de robot que pour le premier scénario (deux robots) ; par contre, leurs positions initiales ont été fixées manuellement.

Nous avons affecté le premier robot à la zone 1 ; ses coordonnées sont (X=11 et Y=6.5). Ensuite, nous avons assigné le deuxième robot à la zone 2 ; ses coordonnées sont (X= 10 et Y=-5.5).

La durée de construction de la carte globale est de 288 secondes et sa précision est de 98,61% ; la carte générée est montrée dans la figure suivante :

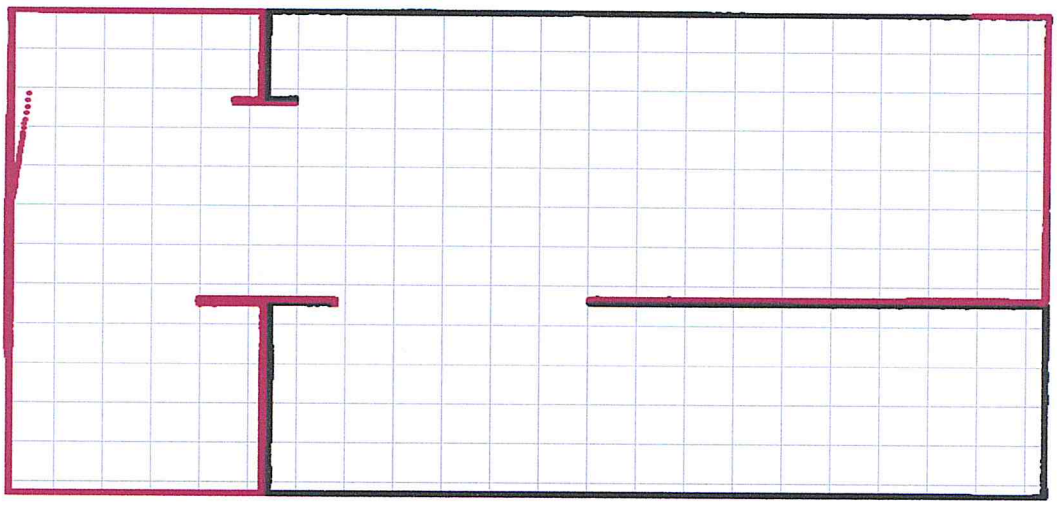


Figure 6 : Carte construite par deux robots (temps de construction de la carte = 288s).

Nous pouvons remarquer que ces cartes globales (premier et deuxième scénario) ne sont pas fidèles à 100% à l'environnement dans lequel évoluent ces deux robots mobiles. Cela est dû essentiellement à la présence d'obstacles dynamiques dans la zone explorée par le deuxième robot (qui consiste en l'autre robot mobile). En effet, le premier robot a été considéré comme étant un obstacle par son homologue.

2.3 Troisième cas : Utilisation de quatre robots mobiles

Nous avons utilisé quatre robots mobiles exploitant l'environnement, capable de le cartographier de façon collaborative.

2.3.1 Premier scénario : Déploiement aléatoire des quatre robots mobiles

Les positions initiales des quatre robots sont générées d'une manière aléatoire, leurs positions respectives sont (X=1 et Y=0), (X=10 et Y=5.3), (X=15 et Y=-5), (X=-13et Y=8.5).

La figure suivante montre les quatre robots entrain d'explorer l'environnement.

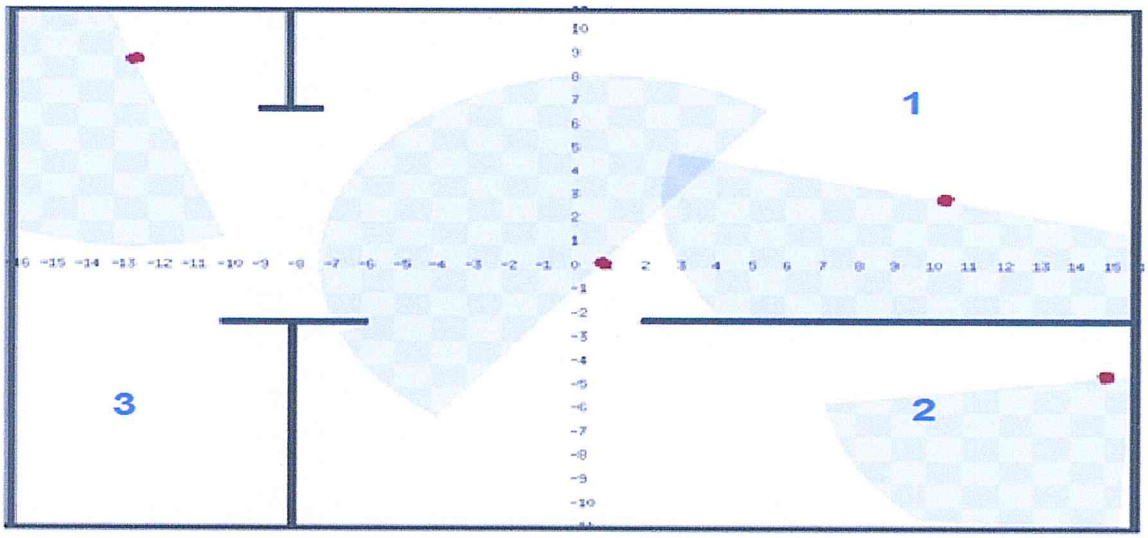


Figure 7 : Environnement exploré par quatre robots

La construction de la totalité de la carte par les quatre robots a durée 144 secondes et sa précision est de 83,2 ; cette carte globale est montrée dans la figure suivante :

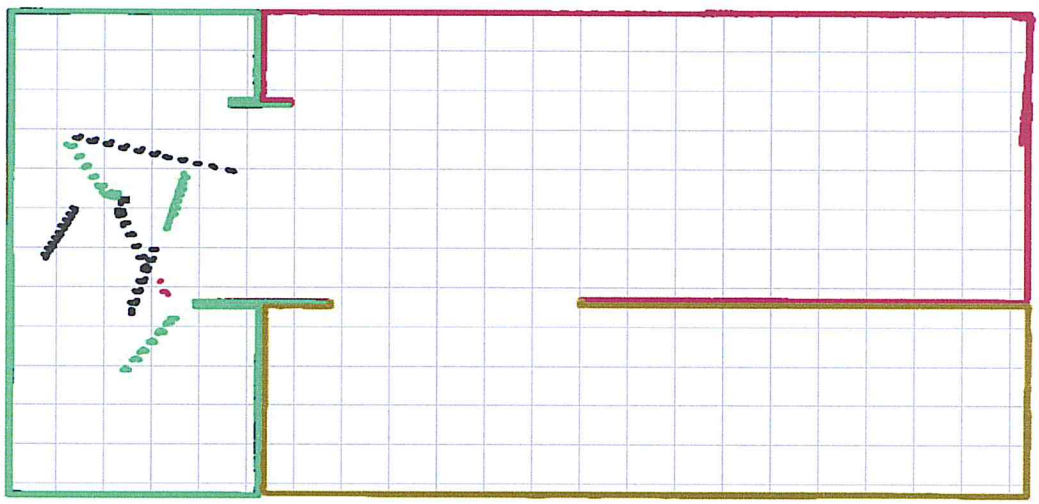


Figure 8 : Carte construite par quatre robots (temps de construction de la carte =144s)

2.3.2 Deuxième scénario : Déploiement manuel des quatre robots mobiles

Dans ce scénario, nous avons choisi le même nombre de robots que le scénario précédent ; mais leurs positions initiales sont déterminées manuellement.

Nous avons affecté le premier robot à la zone 1 ; ses coordonnées sont (X=9.5 et Y=5).Ensuite, le deuxième robot a été affecté à la zone 2 ; ses coordonnées sont (X= 9 et Y=-7). Enfin, le troisième et quatrième robot sont déployés au niveau de la zone 3 ; les

coordonnées du troisième robot sont (X= -10 et Y=-8)et les coordonnées du quatrième robot sont (X=-11.5 et Y=4.5).

La durée de la construction de la carte est de 116 secondes et sa précision est de 84%. La figure suivante montre la carte construite.

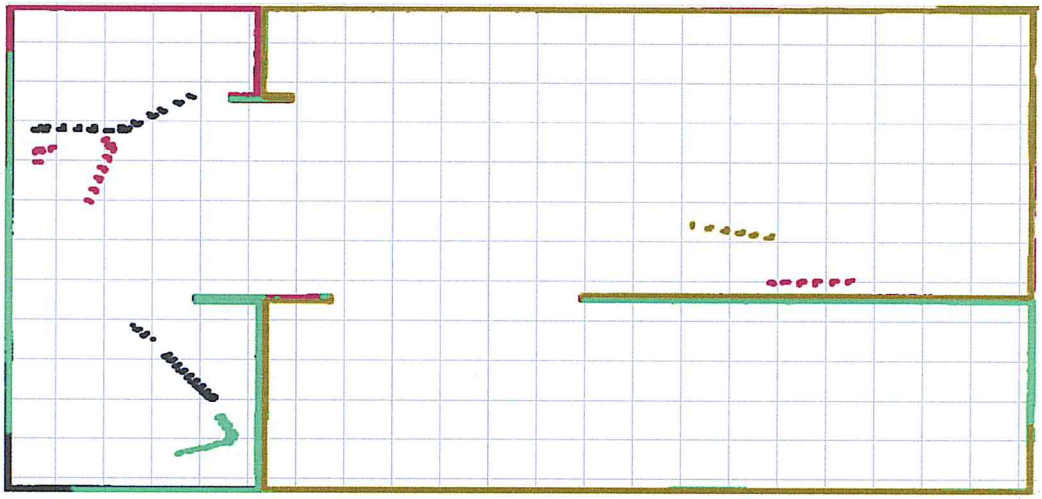


Figure 9 : Carte construite par les quatre robots (temps de construction de la carte = 116s)

Dans ce cas aussi, nous remarquons que les taux d'erreur de construction des deux cartes globales sont plus élevés. Comme pour le deuxième cas, ces erreurs sont dues essentiellement à la présence de plusieurs robots mobiles (obstacles dynamiques) dans la zone explorée par un des robots. En effet, chaque robot mobile considère son homologue comme étant un obstacle à cartographier.

3. Validation dans un environnement structuré complexe

Ces simulations consistent en l'exploration et la cartographie d'un environnement d'intérieur inconnu et structuré en utilisant le même type de robots et les mêmes scénarios précédents. La différence par rapport aux précédents tests consiste en la complexité supérieure de l'environnement considéré.

3.1 Premier cas : Utilisation d'un seul robot mobile

Nous avons utilisé un seul robot mobile qui se déplace de façon aléatoire dans son environnement et qui doit le cartographier entièrement.

3.1.1 Premier scénario : Déploiement aléatoire du robot mobile

La position du robot est générée de façon aléatoire (X=9 et Y=0). La figure suivante montre le robot entrain d'explorer son environnement.

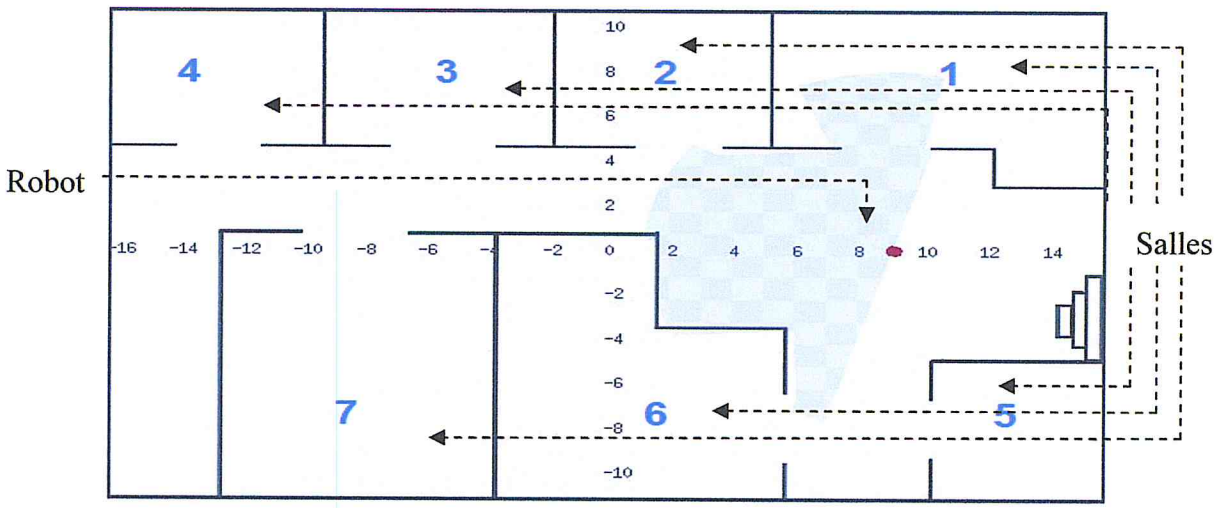


Figure10 : Environnement à explorer par un robot mobile

La construction de la carte globale par un seul robot mobile a durée 720 secondes et sa précision est de 100%; cette carte est donnée par la figure suivante :

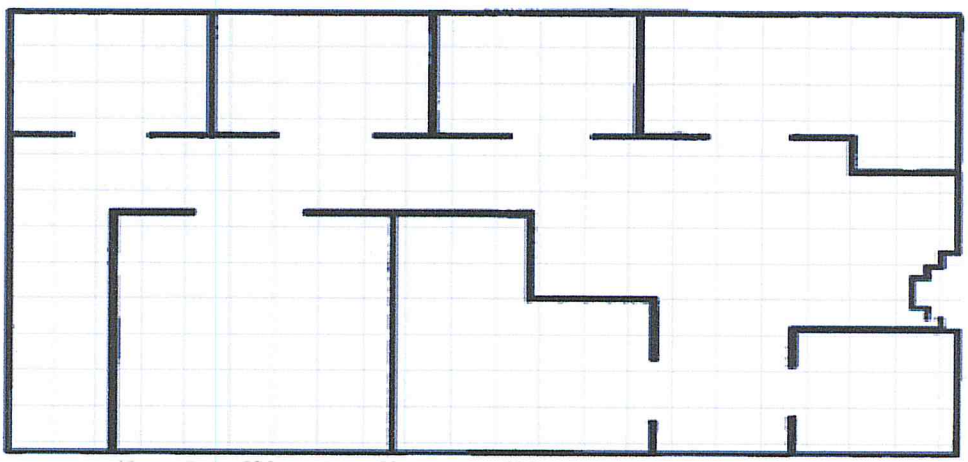


Figure 11 : Carte construite par un seul robot mobile (temps de construction de la carte = 720s)

3.1.2 Deuxième scénario : Déploiement manuel du robot mobile

Il consiste toujours à utiliser un seul robot mobile, mais cette fois-ci, sa position initiale est fixée manuellement.

Nous avons affecté le robot à la salle 1 ;ses coordonnées initiales sont (X=10 et Y=6.5). La durée de la construction de la carte globale est de810 secondes et sa précision est de 100%. Cette carte est montrée par la figure suivante :

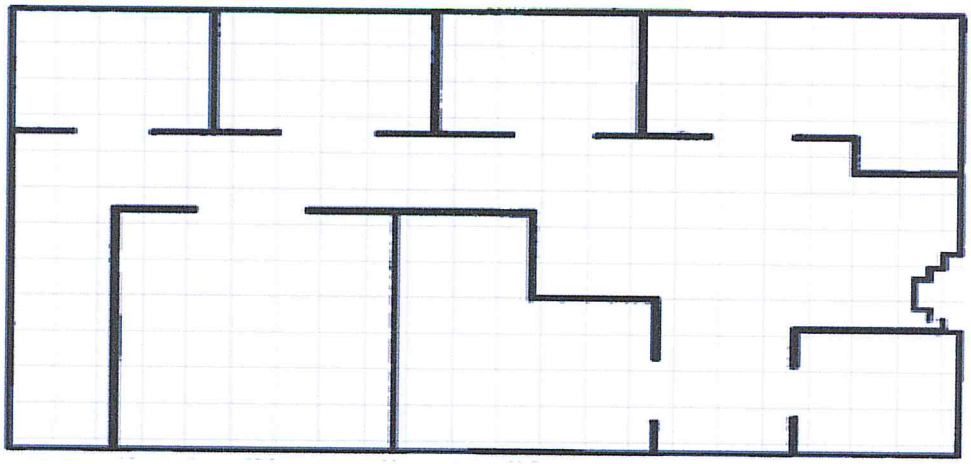


Figure12 : Carte globale construite par un seul robot mobile (temps de construction de la carte = 810s)

3.2 Deuxième cas : Utilisation de deux robots mobiles

Dans ce cas, nous avons considéré deux robots mobiles.

3.2.1. Premier cas : Déploiement aléatoire des deux robots mobiles

Les positions initiales des deux robots sont générées d'une manière aléatoire. La figure suivante montre les deux robots entrain d'explorer l'environnement.

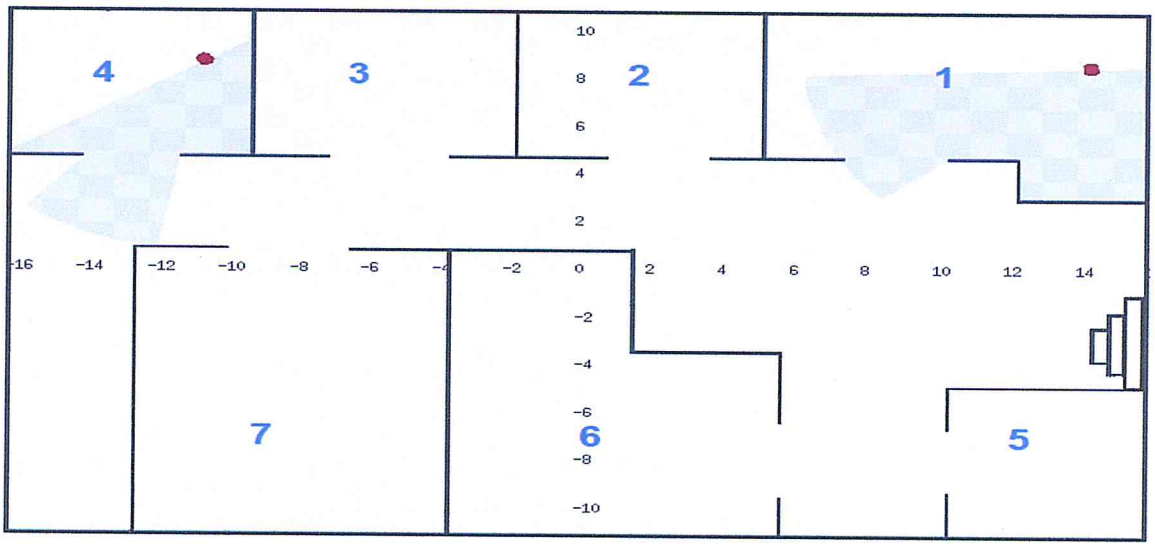


Figure 13 : Deux robots mobiles entrain d'explorer leur environnement

La durée de construction de la carte globale est de 560 secondes et sa précision est de 97%;elle est représentée ci-dessus (figure 14) :

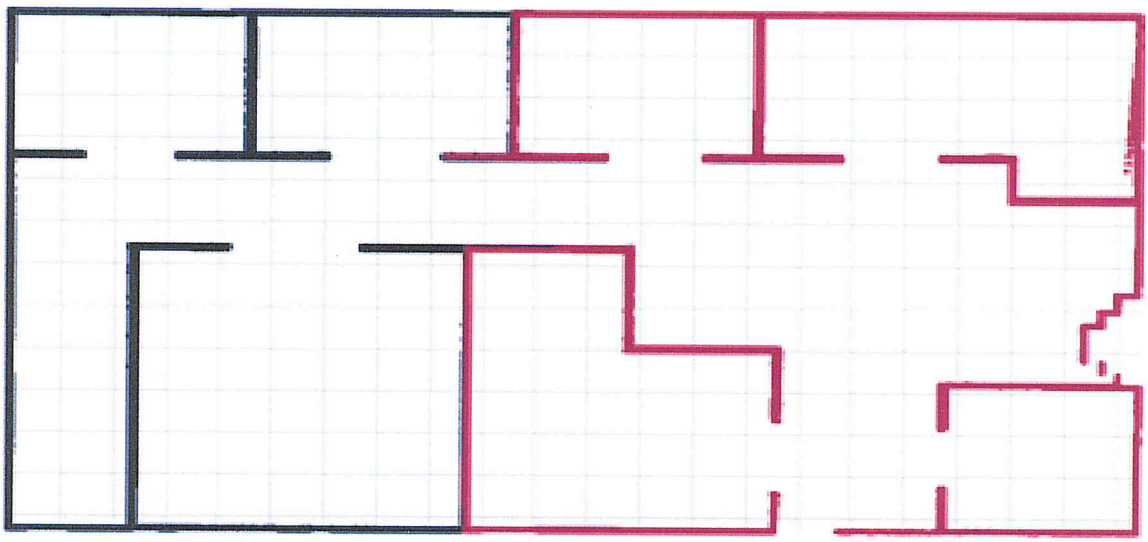


Figure14 : Carte construite par les deux robots mobiles (temps de construction de la carte = 560s)

3.2.2 Deuxième scénario : Déploiement manuel des deux robots mobiles

Nous avons affecté le premier robot à la salle 2 ; ses coordonnées sont (X=3.5 et Y=7). Le deuxième robot a été déployé au niveau de la salle 7 ; ses coordonnées sont (X=-5.5 et Y=-3).

La carte globale a été construite en un temps de525 secondes et sa précision est de 99,04% ; elle est montrée dans la figure 15 suivante:

Figure15 : Carte globale construite par deux robots(temps de construction de la carte = 525s).

Nous remarquons, dans ce cas aussi, que des erreurs de cartographie existent au niveau des zones explorées par le premier et le deuxième robot. En effet, le premier robot a exploré une partie de la zone 2, et le deuxième robot a exploré une partie de la zone 1. En chaque zone, le premier robot mobile considère l'autre robot comme étant un obstacle qu'il faut cartographier et vice-versa.

3.3 Troisième cas : Utilisation de quatre robots mobiles

Dans ce dernier cas, nous avons utilisé quatre robots mobiles pour l'exploration et la cartographie de l'environnement :

3.3.1 Premier scénario : Déploiement aléatoire des quatre robots mobiles

Les positions initiales des quatre robots mobiles sont générées d'une manière aléatoire. La figure suivante les montre en positions initiales entrain d'explorer l'environnement.

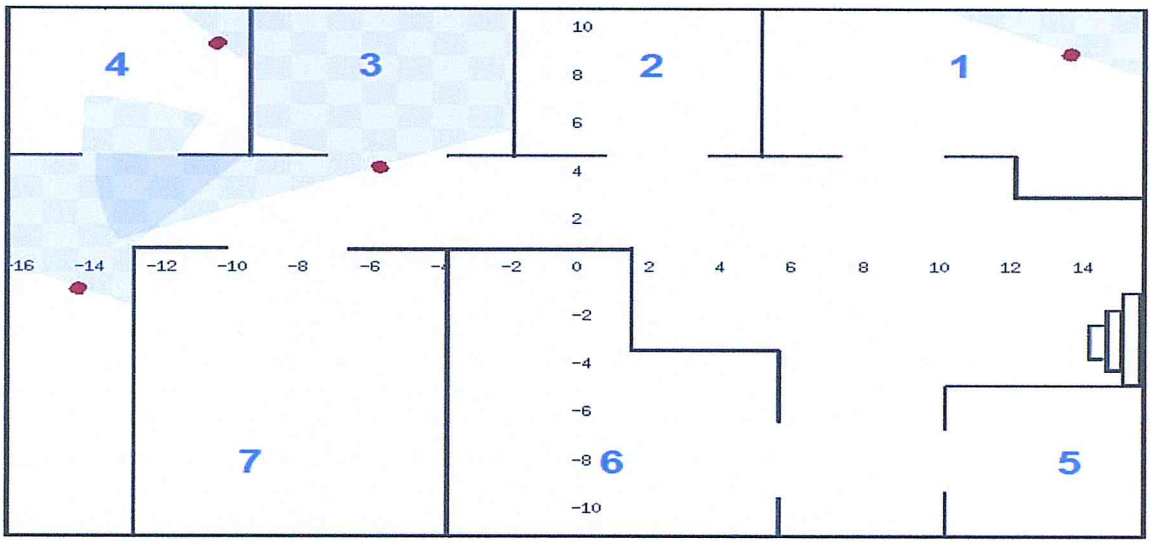


Figure 16 : Environnement à explorer par les quatre robots mobiles

La construction de la carte globale a durée 310 secondes et sa précision est de 83% ; elle est donnée dans la figure suivante :

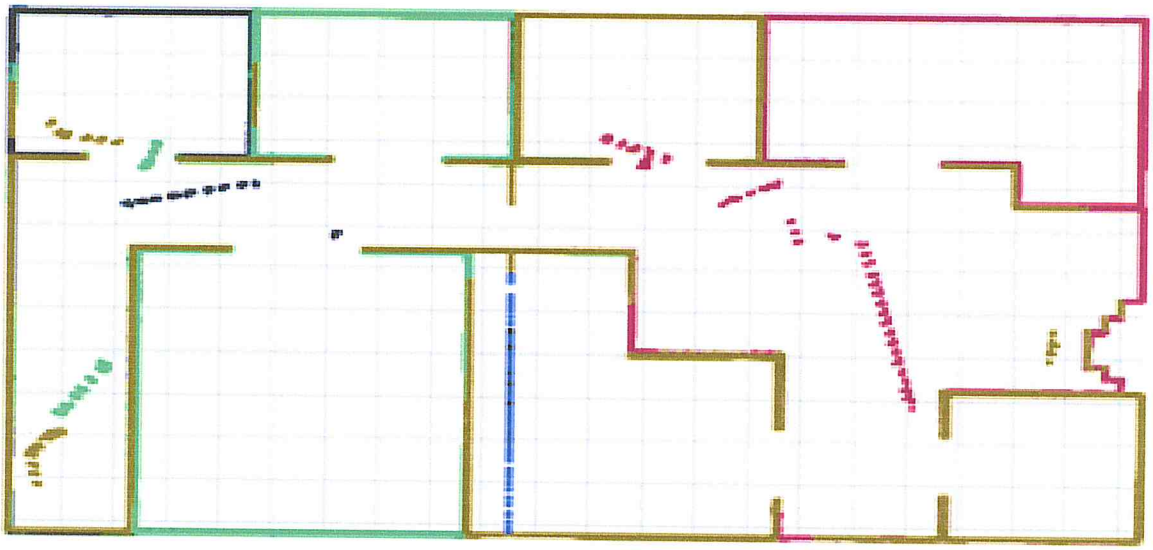


Figure17 : Carte construite par quatre robots (temps de construction de la carte = 310s).

3.3.2 Deuxième scénario : Déploiement manuel des quatre robots mobiles

Nous avons affecté le premier robot mobile à la salle 1 ; ses coordonnées sont (X=11 et Y=8).Le deuxième robot a été déployé au niveau de la salle 4 ; ses coordonnées sont (X=-9.5 et Y=8).Le troisième robot a été affecté à la salle 5 ; ses coordonnées sont (X=13 et Y=-8).Enfin, le quatrième et dernier robot a été assigné à la salle 7 ; ses coordonnées sont (X=-4 et Y=-6.5).

La construction de la carte globale a durée219 secondes et sa précision est de 90% ; elle est donnée dans la figure suivante :

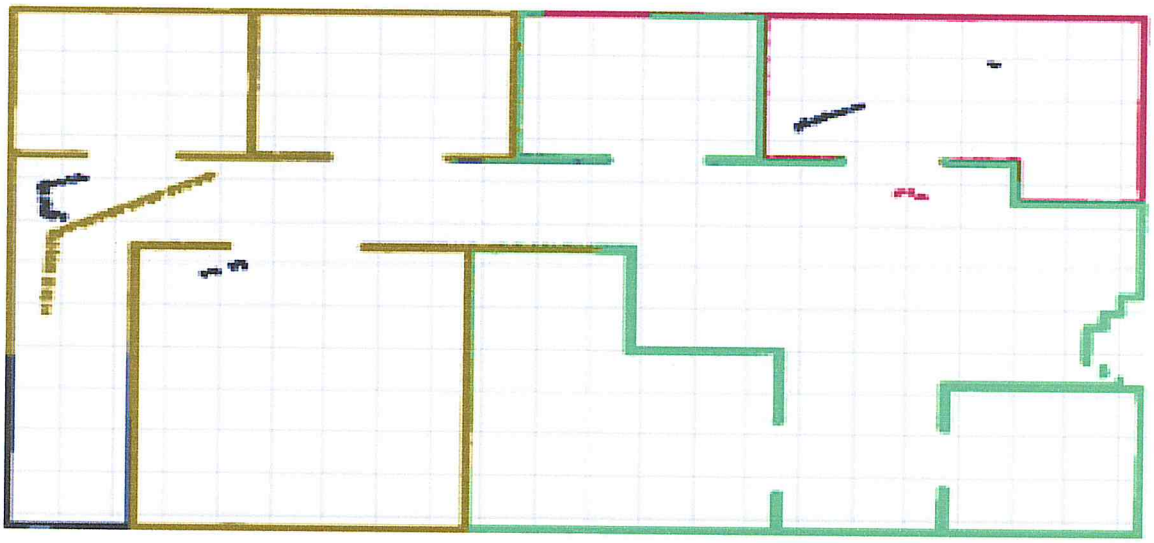


Figure18 : Carte construite par quatre robots (temps de construction de la carte = 219s).

L'utilisation de quatre robots mobiles pour explorer et cartographier l'environnement dans lequel évoluent ces robots a permis de minimiser considérablement le temps total de construction de la carte globale. Malheureusement, les précisions des deux cartes ne sont pas bonnes. En effet, la détection mutuelle des robots mobiles (considérés comme des obstacles dynamiques) a diminué considérablement la qualité des cartes finales générées et a créé des obstacles fictifs qui n'existent pas dans l'environnement réel.

4. Discussion des résultats obtenus

Le tableau 1 résume les temps nécessaires pour la construction des cartes globales dans les différents scénarios présentés précédemment.

Nombre de robots	Environnement structuré simple		Environnement structuré complexe	
	Déploiement aléatoire	Déploiement manuelle	Déploiement aléatoire	Déploiement manuelle
1	496s	540s	720s	810s
2	324s	288s	560s	525s
4	144s	116s	310s	219s

Tableau 1 : Temps de construction des cartes globales pour les environnements structurés simple et complexe

Le tableau 2 suivant résume la précision des cartes globales généréesdans tous les cas considérés dans ce chapitre.

Nombre de robots	Environnement structuré simple		Environnement structuré complexe	
	Déploiement aléatoire	Déploiement manuelle	Déploiement aléatoire	Déploiement manuelle
1	100%	100%	100%	100%
2	94%	98.61%	97%	99.04%
4	83.2%	84%	83%	90%

Tableau 2 : Précision des cartes globales générées pour les environnements structurés simple et complexe

Les résultats de construction collaborative de cartes pour les environnements simple et complexe considérés dans ce travail nous ont permis de tirer les conclusions suivantes :

- La construction d'une carte avec un seul robot mobile prend plus de temps qu'avec un groupe de robots.
- L'utilisation d'une bonne stratégie de déploiement de robots mobiles permet de minimiser le temps total de construction de la carte globale.
- Les robots prennent plus de temps à cartographier un environnement plus complexe.
- L'utilisation d'un groupe de robots mobiles augmente le taux d'erreurs et minimise la précision de la carte globale générée. Car si un robot mobile croise un autre robot, il le cartographie en le considérant comme un obstacle faisant partie de l'environnement.

6. Conclusion

Tout au long de ce chapitre, divers scénarios ont été réalisés avec le simulateur Player/Stage. En examinant les résultats obtenus, nous avons pu en déduire les avantages et les inconvénients d'un système multi-robots dédié à la construction collaborative de carte de l'environnement.

L'avantage principal de cette collaboration est que les robots permettent de cartographier plus rapidement l'environnement en utilisant une bonne stratégie de déploiement (stratégie manuelle) par rapport à un système mono-robot. Par ailleurs, l'approche multi-robots comporte aussi des inconvénients, par exemple les robots peuvent parcourir les mêmes zones et même entrer en collision ; aussi, un robot mobile détectant un autre peut le considérer comme étant un obstacle faisant partie de l'environnement.

Néanmoins, nous pouvons facilement pallier à ce dernier inconvénient en dotant le système multi-robots de la capacité à partager des informations sur les positions actuelles des robots. À cet effet, chaque fois qu'un robot détecte un obstacle, il compare sa position avec celles de tous les robots mobiles évoluant dans son voisinage.

Conclusion générale et perspectives

Conclusion générale et perspectives

Dans le domaine de la robotique mobile, l'une des tâches les plus stratégiques et importantes concerne l'exploration et la cartographie d'un environnement physique. Elle consiste, dans les cas les plus difficiles, à découvrir un environnement inconnu ou ayant subi des modifications et à en construire une représentation.

Les cartes qui en résultent sont souvent essentiellement nécessaires aux robots mobiles pour effectuer des tâches complexes comme la recherche et le secours de victimes après certaines catastrophes naturelles (tremblement de terre, incendie, etc.) ainsi que l'inspection de zones dangereuses comme les zones radioactives. Ces cartes sont aussi essentielles aux êtres humains afin de reconnaître des environnements lointains, des environnements inaccessibles ou encore hostiles.

Cependant, l'utilisation d'un groupe de robots contribue à l'accomplissement de la tâche de la cartographie de manière collaborative reste, jusqu'à présent, un problème difficile. Car pour réaliser un modèle cohérant d'un environnement quelconque, une exploration complète devrait être effectuée par le groupe de robots ainsi que les données recueillies devraient être intégrées dans une seule carte globale.

L'objectif de notre travail consiste en la construction collaborative d'une carte par un groupe de robots mobiles et homogènes. Ce travail commence par assigner chaque robot à une zone à explorer afin de récupérer toutes les informations nécessaires et les cartographier en parallèle, et par la suite, la flotte de robots communique ces informations sur toutes les parties de l'environnement à un serveur central de carte. L'objectif étant aussi d'explorer plus rapidement la totalité de l'environnement ainsi que la minimisation du temps total nécessaire à la construction de la carte.

Dans ce mémoire, nous avons parlé des différentes stratégies d'explorations existantes en les introduisant selon différents objectifs (Minimisation du temps d'exploration, Maximisation de la précision, etc.).Nous avons ainsi abordé les différentes approches d'identification des zones à explorer en outre des différentes approches d'affectation.

Nous avons, par la suite, présenté les différentes stratégies de coordination qui se réfère à l'équilibre des robots selon différentes architectures (architecture centralisée, architecture décentralisée, etc.). Nous avons aussi décrit les principaux types de représentation qui reflète l'architecture de l'environnement à partir des différentes informations recueillies par les robots (carte métrique, carte topologie, etc.).Nous avons, en outre, présenté divers exemples ainsi que les travaux les plus réputés dans le domaine de l'exploration et la cartographie multi-robots.

L'implémentation de l'approche de construction collaborative de cartes de l'environnement a été réalisée avec le simulateur Player/Stage. Ce simulateur permet de simuler des environnements (encombré ou libre), et aussi de simuler plusieurs robots équipés de capteurs laser. Ce capteur a été aussi utilisé pour la récolte des données (obstacles, etc.) dans le but de faire la cartographie.

Nous avons choisi l'architecture centralisée comme stratégie de coordination afin que les robots puissent communiquer au serveur central de cartes toutes les données nécessaires à la cartographie. Ce type d'architectures centralisée est très efficace pour un petit nombre de robots et facile ; elle est facile à mettre en œuvre et permet de fournir une vue globale du monde dans lequel évoluent les robots mobiles.

Le type de représentation de l'environnement choisi a été la carte métrique ; plus précisément, la carte géométrique. Ce type de cartes offre une représentation meilleure et précise de l'environnement ; de plus, elles sont mieux adaptées aux environnements d'intérieurs.

Nous avons implémenté un algorithme de cartographie qui se base sur la détection des limites de l'environnement et des obstacles ainsi que leurs positions par rapport aux robots. Ceci nous a facilité énormément le processus de construction des cartes et a permis d'augmenter leurs précisions.

Enfin, nous avons évalué en simulation les performances de l'algorithme utilisé via plusieurs scénarios en termes de temps nécessaire lors de la construction de la carte globale et en termes de précision de la carté générée.

Nous avons constaté que l'utilisation d'un groupe de robots mobiles déployés selon une bonne stratégie a permis de cartographier plus rapidement l'environnement, c'est-à-dire la construction de la carte globale se fait en un temps minimum. Nous avons aussi remarqué que l'utilisation d'un groupe de robots mobiles est une approche robuste ; car si un robot tombe en panne ou il est bloqué contre des obstacles, cela ne signifie pas l'arrêt total du système de la

cartographie ; bien au contraire, les autre robots peuvent continuer cette tâche assurant ainsi un service minimum qui est un aspect très intéressant en qualité de service.

Durant la réalisation de notre travail, nous avons été confrontés à plusieurs problèmes avec le simulateur Player/Stage. L'indisponibilité d'un guide complet du simulateur (étape par étape) nous a obligé à consacré beaucoup de temps et d'efforts afin d'apprendre à le maîtriser. De plus, son installation sous le système d'exploitation linux était très complexe. Aussi, Player/Stage a un grand nombre de dépendances requises pour pouvoir fonctionner et s'exécuter de façon optimale.

L'algorithme de cartographie présenté dans ce mémoire traite seulement les environnements statiques, l'extension de cet algorithme aux environnements dynamiques (situations réelles) améliorera considérablement ses performances. Le problème des environnements dynamiques est lié aux problèmes des erreurs et de contradiction structurale dans la carte. Car dans la cartographie, il est nécessaire de distinguer les objets statiques des objets dynamiques afin de ne pas intégrer les objets dynamiques dans la carte ce qui engendre une représentation erronée.

Il serait aussi plus intéressant d'étendre ce travail à l'exploration des environnements en trois dimensions qui est un problème également d'actualité en robotique. Pour cela Gazebo pourrait être utilisé. Celui-ci est une simulation dynamique conçue pour la simulation des robots dans des environnements en 3D.

Enfin, il est intéressant de tester les performances de notre travail sur des robots réels équipés de capteurs dans divers environnements.

Références bibliographiques

[1] I. Bouyoucef, " Coordination de robots pour le transport d'objets", Master 2Systèmes complexes Technologies de l'Information et du Contrôle (ScTIC), Université Paris Est Créteil, France, 2013, pp. 4-15.

[2] Z. Yan, "Contributions à la coordination de tâches et de mouvements pour un système multi-robots ", Doctorat de l'Université Paris 8, Spécialité Informatique, France, 2012, pp.1-20.

[3]M. Mouad, "Architecture de COntrole/COmmande dédiée aux systèmes Distribués, autonomes (ACO2DA) : application à une plate-forme multi-véhicules", Université Blaise Pascal - Clermont-Ferrand II, France, 2014, pp. 28.

[4] J. Beaudry, "Machine décisionnelle pour système multi-robots coopératifs ",École Polytechnique de Montréal, France, 2005, pp. 2.

[5] V. Tuan Le, " Coopération dans les systèmes multi-robots : Contribution au maintien de la connectivité et à l'allocation dynamique de rôles", université de Caen, basse Normandie -UFR des sciences, France, 2010, pp. 5-6.

[6] J. Spiewak, "Contribution à la coordination de flottille de véhicules sous-marins autonomes", Université Montpellier II - Sciences et Techniques du Languedoc, France, 2007, pp. 89-100.

[7] B. Rivollet, C.E. Serre, "Les drones pour la surveillance environnementale", France, 2014, pp.5.

[8] R. Drouilly, "SLAM Visuel 3D pour robot mobile autonome", Université de Strasbourg, France, 2011, pp.9-10.

[9] I. Andersone, "The Characteristics of the Map Merging Methods: A Survey",Scientific Journal of Riga Technical University, vol. 43, 2010.

[10] A. Bautin, "Stratégie d'exploration multi-robots fondée sur le calcul de champs de potentiels", Laboratoire Lorrain de Recherche en Informatique et ses Applications,Lorraine, 2013, pp. 10-24.

[11] A. Marjovi, J.G. Nunes, L. Marques, A.de Almeida, " Multi-robot exploration and fire searching", In Proceedings of the 2009 IEEE/RSJ International Conference on

Intelligent Robots and Systems (IROS'09),St. Louis, MO, USA, Octobre 2009, pp. 1929-1934.

[12] Y. Mei, Y. -H.Lu, C. S. G. Lee,Y. C. Hu,"Energy-efficient mobile robot exploration", In Proceedings 2006 IEEE International Conference on Robotics and Automation, May 2006, pp. 505-5011.

[13]A.A. Makarenko , S. B. Williams, F. Bourgault, H. F. Durrant-Whyte,"An experiment in integrated exploration", In Proceedings 2002. IEEE/RSJ International Conference on Intelligent Robots and Systems,vol. 1,Australia, 2002, pp. 534 - 539.

[14] C. Stachniss, D.H ahnel, W. Burgard, G. Grisetti, "On Actively Closing Loops in Grid-based FastSLAM",Advanced Robotics, vol. 19,,Italy, 02 Avril 2012,pp. 1059-1079.

[15]G. Lozenguez, "Stratégie coopérative pour la mise en œuvre d'une flotte de robots mobiles dans un milieu ouvert et encombré ", université de Caen, Basse Normandie,France, 2012, pp.23.

[16] L.E. Parker, F.E. Schneider,A.C. Schultz, "Multi-Robot Systems.FromSwarms to Intelligent Automata",Proceedings from the 2005 International Workshop on Multi-Robot Systems, vol. 3, USA, 2005.

[17] H. Ghazouani, "Navigation visuelle de robots mobiles dans un environnement d'intérieur",Robotics Université Montpellier II - Sciences et Techniques du Languedoc, France, 2012,pp. 26-29.

[18] T. Bailey "Mobile Robot Localization and Mapping in Extensive Outdoor Environments". Australian Centre for Field Robotics Department of Aerospace, Mechanical and Mechatronic Engineering the University of Sydney, Australia, Aout 2002, pp. 13-25.

[19] O. Maler, " Navigation Multi-Robots En Milieu Urbain Dynamique", Rapport de Master IVR INPG, UJF, France,16 Juin 2007, pp. 6-7.

[20] K. Singh, K. Fujimura, "Map Making by Cooperating Mobile Robots". In Proceedings, 1993 IEEE International Conference onRobotics and Automation, vol. 2, May 1993, pp. 254-259.

[21] B. Yamauchi, "Frontier-Based Exploration Using Multiple Robots", In Proceedings of the Second International Conference on Autonomous Agents (Agents '98), Minneapolis, MN, Washington, Mai 1998, pp. 1-8.

[22] R.Zlot, A.T. Stentz, M. B. Dias, S. Thayer, "Multi-robot exploration controlled by a market economy", In Proceedings of the 2002 IEEE International Conference on

Robotics and Automation (ICRA'02), Washington, DC, USA, May 2002 pp. 3016-2023.

[23] K.M. Wurm, C. Stachniss, W. Burgard, "Coordinated multi-robot exploration using a segmentation of the environment", In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08),Nice, France, Septembre 2008, pp. 1160-1165.

[24] B.P. Gerkey a, M.J. Mataric, "Auction methods for multi-robot coordination", In Proceedings of the 2002 IEEE Transactions on Robotics and Automation, Octobre 2002, pp. 758-768.

[25] J.G. Rogers, C. Nieto-Granda, H.I. Christensen, "Coordination strategies for multi-robot exploration and mapping", Center for Robotics and Intelligent Machine, Georgia Institute of Technology, Vol. 88, 2013, pp. 231-243.

[26] W. Burgard, M. Moorsy, C. Stachniss, F. Schneider, "Coordinated Multi-Robot Exploration", Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany,2005, pp. 2-5.

[27] N.M. Cuong, "Adotion de formations spatiales d'un groupe de robot à partir du Wifi ", Institut de la francophonie pour l'informatique, France, 2009, pp. 6-8.

[28] K.M. Wurm, C. Stachniss, W. Burgard, "Coordinated Multi-Robot Exploration using a Segmentation of the Environment", In Proceedings of IROS 2008. IEEE/RSJ International Conference onIntelligent Robots and Systems, Septembre 2008, pp. 1160-1165.

[29] D. Filliat, "Adoption de formations spatiales d'un groupe de robots à partir du Wifi", Institut de la Francophonie pour l'informatique, École Nationale Supérieure de Techniques Avancées Paris Tech, France, 2012, pp. 19-102.

[30] S.C. Botelho, A. lamiR, M+ : "A scheme for multi-robot cooperation through negotiated task allocation and achievement ", In Proceedings of the 1999 IEEE International Conference on Robotics and Automation, USA, Mai 1999, pp. 1234-1239.

[31]T. Fukuda, T. Ueyama, Y. Kawauchi, F. Arai, "Concept of cellular robotic system (CEBOT) and basic strategies for its realization",Computers et Electrical EngineeringVol. 18,Janvier 1992, pp. 11-39.

[32] R. Madhavan, K.Fregene, L.E. Parker, " Distributed Heterogeneous Outdoor Multi-robot Localization", In Proceedings. ICRA'02. IEEE International Conference on Robotics and Automation, vol. 1, USA, 2002, pp. 374 - 381.

[33] B.P.Gerkey,M. J. Matari´c, " Principled communication for dynamic multi-robot task allocation", In Proceedings of the 7th International Symposium On Experimental Robotics (ISER'00), Honolulu, HI, USA, Decembre 2000, pp. 353-362.

[34] G. Hollinger, S. Singh, J. Djugash, A. Kehagias, "Efficient Multi-Robot Search for a Moving Target",The International Journal of Robotics Research, vol. 28, February 2009, pp. 201-219.

[35] T. J. Koo,S. M. Shahruz, "Formation of a group of unmanned aerial vehicles(UAVs) ", In Proceedings of the 2001 on American Control Conference, vol. 1,USA, juin 2001, pp. 69-74.

[36] A. Baba, "Cartographie de l'environnement et suivi simultané de cibles dynamiques arun robot mobile ",Automatic. Université Paul Sabatier - Toulouse III, France, 2007, pp. 9-16.

[37] J. A. Castellanos, J. M. M. Montiel, J. Neira, D. Tardos, "The spmap: a probabilistic framework for simultaneous localization and map building", In IEEE Transactions on Robotics and Automation, vol. 15, octobre 1999, pp. 948 - 952.

[38] J. A. Janet, S. M. Scoggins, M. W. White, J. C. Sutton, E. Grant, W. E. Snyder, "Self-organising geometric certainty maps: A compact and multifunctional approach to place recognition and motion planning" In Proc.IEEE Intl. Conf. on Robotics and Automation, volume 4, 1997.

[39] D. Nour El Islam, "Localisation et navigation des robots Mobiles 'LNRM'", Université d'Oran, Algérie, pp. 5-10.

[40] A. Alfes, "Sonar-based Real-world Mapping and Navigation" In IEEE Journal of Robotics and Automation, vol. 3, USA, June 1987, pp. 249-265.

[41] A. Birk, S. Carpin, "Merging occupancy grid maps from multiple robots", In Proceedings of the IEEE, vol. 94, Juillet 2006, pp. 1384-1397.

[42] P. Stepan, M. Kulich, L. Pireucil. "Robust data fusion with occupancy grid", IEEE Transactions on systems, MAN, and Cybernetics-Part C: Applications and Reviews, Vol 35, N° 1, 2005.

[43] E. Fabrizi, A.Saffiotti," Augmenting Topology-Based Maps with Geometric Information", In Robotics and autonoums SystemsRoma, vol. 40, Italy,2002, pp. 91-97.

[44] B. Kuipers, Y. -T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations", Robotics and autonoums Systems Texas, vol. 8, USA,1991, pp.47-63.

[45] D.Kortenkamp, T. Weymouth, "Topological mapping for mobile robot using a combination of sonar and vision sensing", In AAAI'94 Proceedings of the twelfth national conference on Artificial intelligence, vol. 2, 1994, pp. 979-984.

[46] A. ZUREIKI," Fusion de Données Multi-Capteurs pour la Construction Incrémentale du Modèle Tridimensionnel Texturé d'un Environnement Intérieur par un Robot Mobile ", Université Toulouse III - Paul Sabatier,France, 2008, pp.37.

[47] S. Carpin, "Fast and accurate map merging for multi-robot systems", University of California, Merced, juin 2008, pp. 1-2.

[48] K. L. Ho, P. Newman, "Multiple Map Intersection Detection using Visual Appearance", Oxford University Robotics Research Group, 2005, pp. 2-4.

[49] G. Dedeoglu, G.S. Sukhatme, "Landmark-based matching algorithm for cooperative mapping by autonomous robots", In Distributed Autonomous Robotic Systems 4, Springer, 2000, pp. 251-260.

[50] H. Hajjdiab, R. Laganiere, "Vision-base Multi-Robot Simultaneous Localization and Mapping", Canadian Conference on Computer and Robot vision, 2004, pp. 155-162.

[51] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, B. Stewart, "Distributed multi-robot exploration and mapping," Proceedings of the IEEE, vol. 7,2006, pp. 1325-1339.

[52] S. Carpin, A. Birk, V. Jucikas,"On map merging", Robotics and Autonomous Systems, vol. 1,2005, pp. 1-14.

[53] K. Ishioka, K. Hiraki, Y. Anzai, "Cooperative map generation by heterogeneous autonomous mobile robots," in IJCAI'93 Workshop on Dynamically Interacting Robots, Aout 1993, Chambery, 1993, pp. 57- 67.

[54] R. Simmons, D.Apfelbaum, W.Burgard,D.Fox, M.Moors, S.Thrun, H.Younes "Coordination for multi-robot exploration and mapping",In AAAI National Conference on Artificial Intelligence, Austin, Aout 2000.

[55] D. Rodriguez-Losada, F. Matia, A. Jimenez, "Local maps fusion for real time multi-robot indoor simultaneous localization and mapping," in IEEE International Conference on Robotics and Automation, vol. 2,New Orleans, Avril 2004, pp. 1308-1313.

[56] N. Ceccarelli, M.Di.Marco, A.Garulli, A.Giannitrapani, A.Vicino "Set membership localization and map building for mobile robots",In Systems & Control: Foundations & Applications, Part III, Boston: Birkhauser, 2006, pp. 289-308.

[57] N. Roy,G. Dudek, "Collaborative robot exploration and rendezvous: algorithms, performance bounds and observations," Autonomous Robots, vol. 11, Springer, 2000,pp. 117-136.

[58] S. I. Roumeliotis, G. A. Bekey, "Distributed multirobot localization," In IEEE Transactions on Robotics and Automation, vol. 18, 2002, pp. 781-795.

[59] J. Ko, B.Stewart, D.Fox, K.Konolige, B.Limketkai "A practical decision-theoretic approach to multi-robot mapping and exploration", in IEEE/RSJ International Conference on Intelligent Robots and Systems, Nevada,Octobre 2003, pp. 3232- 3238.

[60] Xin.Ma,R.Guo,L. Yibin, C.Weidong, "Adaptive genetic algorithm for occupancy grid maps merging",In IEEE 7th World Congress onIntelligent Control and Automation, Juin 2008, pp. 5716 -5720.

[61] Brian p.Gerkey, richardt.vaughan, andrewhoward, "most valuable player: a robot device server for distributed control", University of southern California, USA, Novembre 03,2001, pp. 1228-1230.

[62] R.Hedges, K.Stoy,USA,(2008): "Player-Stage".Inhttp://playerstage.sourceforge.net, 25 Avril 2015.

[63]B.Jung, "Player Tutorial",Center for Robotics and Embedded Systems, California, pp.5.

[64] C.Djeutcheu, "Etude et mise en place d'un système informatisé de transfert d'argent inter-agence COMECI ", Université de Dschang-ISMA, Cameroun, 2010, pp. 41-43.

[65] K.S. Senthilkumar,K. K. Bharadwaj, "player-Stage based simulator for simultaneous multi-robot exploration and terrain coverage problem", International Journal of Artificial Intelligence & Applications (IJAIA), vol.2, No.4, Octobre 2011

[66] G.Ceme, M.Garcia, C.González, S.González, " Generation of maps using a Pioneer 2DX mobile robot in a simulated environment Player/Stage", International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS),Mexico, 2014, pp. 17-20.

[67] R.Guyonneau. "Méthodes ensemblistes pour la localisation en robotique mobile",École doctorale : STIM (Science et Technologies de l'Information et Mathématiques), France, 2013, pp. 20-22.

Annexe Player/Stage

1. Introduction a Player/Stage

Le Projet Player/Stage fournit des outils open source qui simplifient le développement de contrôleur, en particulier les systèmes multi-robots .Ce projet a commencé à l'Université de Californie du Sud en 1999 et a déménagé à la source-Forge en 2001 pour répondre à un besoin interne de l'interface et de simulation pour le système multi-robots. Il a ensuite été adopté, modifié et étendu par des chercheurs du monde entier. Player / Stage offre une combinaison de transparence, de flexibilité et de rapidité. Il permet de tester et de développer des algorithmes sans la nécessité d'un environnement réel.

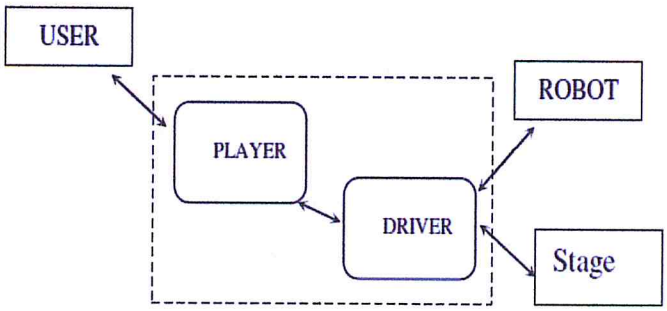


Figure 1 : l'architecture de Player/Stage

2. Player

Player définit un ensemble d'interfaces standard, puissantes et flexibles qui spécifient différentes façons d'interagir avec les dispositifs robotiques. La figure 1 illustre l'architecture du Player / Stage. Il se concentre exclusivement sur les capteurs et actionneurs et il tente d'établir un cadre basé sur le modèle client/serveur pour permettre les communications à base de TCP entre les dispositifs à base de robots. De cette façon, Player agit comme une couche d'abstraction matérielle. Player peut fonctionner sur de nombreux ordinateurs Unix fournissant une interface simple pour les capteurs, les actionneurs et autres dispositifs de robot, Le logiciel contient de nombreuse fonctions puissantes qui appelle a des pilotes non spécifiques permettant au programmeur de réutiliser les programmes de contrôle sur les différents robots sans réécrire le code .Il

existe des interfaces normalisées qui sont appuyées par de nombreux pilotes, un pour chaque type de robot ou capteur.

2.1 Interface

L'interface définit la syntaxe et la sémantique de tous les messages qui peuvent être échangés avec des entités dans la même classe. Player sert d'interface de dispositif de robot. L'interface peut contrôler des vitesses linéaires et angulaires et retransmet des informations d'odométrie. Player définit un ensemble d'interfaces standard, dont chacune est une spécification de la façon dont l'utilisateur peut interagir avec une certaine classe de dispositifs robotisés tels que les capteurs, les actionneurs ou les algorithmes.

Un programme qui utilise ces interfaces est en mesure de contrôler, sans aucune modification ou recompilation, différents types de robots. La seule modification serait dans un fichier de configuration utilisé par le serveur de Player qui contient la déclaration de chaque dispositif et de son pilote respectif.

2.2 Les pilotes et les dispositifs

Le pilote est un logiciel qui prend en charge un matériel spécifique, qui communique avec des capteurs et des actionneurs robotiques. Le rôle du pilote est de rendre le robot soutenable a l'interface standard et à cacher les détails de toute entité donnée par ce qui semble être le même que toute autre entité dans sa catégorie. Par exemple dans le Player, le pilote sicklms200 contrôle SICK LMS200, qui est un capteur de distance de la plupart des pilotes prennent en charge l'interface 2d position, y compris p2os, obot et Rflex, qui commandent chacune différents types de robots. Ainsi, le pilote peut aussi traduire les données récupérées pour les faire correspondre avec le format défini par l'interface laser. En Player, le transfert de tous les messages se produit entre les périphériques, grâce à des interfaces. Par exemple, le conducteur sicklms200 peut créer un dispositif, qui pourrait avoir l'adresse suivante: "localhost: 6665: laser: 0". Les champs de cette adresse correspondent aux entrées de la structure de player_devaddr_t: accueil, robots (numéro de port), l'interface, et l'index. Les champs d'hôte et de robots indiquent où se trouve le dispositif et le champ de l'interface indique quelle interface les périphériques supportent, et comment il peut être utilisée.

Le Player nous permet d'avoir accès à beaucoup de dispositifs simultanément, donc nous avons besoin de spécifier les données ci-dessus pour avoir accès aux dispositifs multiples.

2.2.1 Mécanisme de transport

Player prévoit également des mécanismes de transport qui permettent l'échange de données entre les pilotes et les programmes de contrôle qui s'exécutent sur des machines différentes. Cela signifie, qu'un serveur de Player peut être exécuté en utilisant des sockets et un pilote spécifié pour le matériel en cause, et tout le programme client peut ensuite être exécuté à partir d'un emplacement différent sur le réseau via les prises spécifiées. Le transport le plus couramment utilisé est maintenant un transport

basé sur les sockets TCP client / serveur. Player est exécutée avec un fichier de configuration qui définit les pilotes à instancier et comment les lié a un matériel spécifique. Les pilotes courent dans le Player et le programme de contrôle de l'utilisateur s'exécute en tant que client à ce serveur de Player.

2.2.2 Référentiel de code de logiciel

Actuellement, il ya beaucoup de pilotes abstraits qui ont été développés et utilisés à la place du matériel. La principale utilisation du pilote abstrait consiste à encapsuler des algorithmes de façon à être facilement réutilisés. Par exemple, le pilote d'AMCL est une implémentation de la localisation Monte Carlo adaptative, un algorithme bien connu pour la localisation probabiliste d'un robot mobile. Ce pilote prend en charge l'interface position 2d, de sorte qu'il puisse être utilisé directement à la place d'odométrie, et Player devient une plateforme commune de développement et le code référentiel pour ces algorithmes.

3. Stage

Stage simule une population de robots mobiles, de capteurs et d'objets de l'environnement. Il a deux objectifs :

(i) Permettre le développement rapide des contrôleurs qui finira par conduire des robots réels;

(ii) Permettre l'expérimentation robotique sans accès au matériel réel et environnements. Il a été spécifiquement conçu pour soutenir la recherche dans les systèmes multi-robots.

Lors de la programmation et l'expérimentation avec de nombreux robots, des avantages du développement rapide sont multipliés, et Stage permet des expériences avec de grandes populations de robots qui seraient trop coûteux à acheter et à entretenir. Il ya plusieurs aspects de la conception de Stage qui le rend approprié pour les systèmes multi-robots:

3.1 Modèle fidèle

Stage fournit des modèles assez simples, des modèles de calculs à bas prix de lots de dispositifs plutôt que de tenter d'imiter tout dispositif avec une grande fidélité. La simulation à basse fidélité peut effectivement être un avantage lors de la conception des contrôleurs de robots qui doivent fonctionner sur les vrais robots, car elle encourage l'utilisation de techniques de contrôle robustes. Faibles exigences de calcul, signifient pouvoir simuler de nombreux appareils sur du matériel de base.

3.2 Graduation linéaire avec la population

Tous les modèles de capteurs utilisent des algorithmes qui sont indépendants de la taille de la population. Ainsi l'exigence de calcul de Stage croît linéairement avec une population.

3.3 Modèles de périphériques configurables, composables

Différents capteurs et actionneurs sont fournis, y compris les sonars, télémètres laser, segmenteurs visuelles de couleur, détecteurs repères. Les modèles sont souvent plus générales et plus souple que toute pièce de matériel spécifique, de sorte que chaque modèle est configuré pour se rapprocher du (réel ou imaginaire) périphérique cible.

3.4 Interface du Player

Tous les modèles de capteurs et d'actionneurs sont disponibles par le biais d'interfaces standard du Player. Généralement, les clients ne peuvent pas faire la différence entre les dispositifs de robots réels et leurs équivalents simulés de Stage. Ainsi Stage hérite de la flexibilité du Player, a une plateforme et une interface neutre pour tous ses appareils.

4. Installation Player/Stage

L'ordre d'installation est important. D'abord, les bibliothèques du système doivent être à jour. Ensuite, les bibliothèques de dépendances pour Player/Stage doivent être téléchargés et installés. Une fois que toutes les bibliothèques sont à jour et correctement installé on installe Player ensuite Stage.

Pour mettre à jour le système à la dernière version on exécute les commandes suivantes dans le terminal :

```
$ sudo apt-get update

$ sudo apt-get upgrade
```

La meilleure façon d'installer les bibliothèques de dépendance est d'utiliser les ".markings" fichiers. Le lien pour les fichiers ".markings" est le suivant:

http://playerstage.sourceforge.net/wiki/Install_ubuntu_packages

Tous les fichiers ".markings" devront être installés pour le Player et Stage. Il est préférable d'installer toutes les dépendances afin que l'installation de Player/Stage se passe bien.

La version du Player qu'on a utilisé est 3.0.2 et la version du Stage est 3.2.2.

Player doit être installé avant Stage (Stage va chercher Player lors de son installation). Si Player n'est pas trouvé, des erreurs se produisent et ils ne peuvent être fixés avec la désinstallation de stage.

6.1 Installation de Player

Pour commencer le processus d'installation, on exécute les commandes suivantes dans un terminal:

```
$ cd player-<3.0.2>
```

$ mkdir build

$ cd build

$ cmake ../

Lorsque la configuration s'est terminée, sans erreurs, on installe Player avec:

$ make

$ sudo make install

Pour tester si Player s'est installé correctement, on tape la commande suivante :

$ player

Une fois que l'installation s'est terminée sans erreurs, Stage peut être installé

6.1 Installation de Stage

Pour commencer le processus d'installation de Stage, on exécute les commandes suivantes dans un terminal:

$ cd Stage- <3.2.2>

$ mkdir build

$ cd build

$ cmake ../

Lorsque la configuration s'est terminée, sans erreurs, on commence à installer Stage avec:

$ make

$ sudo make install

Une fois que Player et Stage soient installés avec succès, une nouvelle fenêtre devrait s'ouvrir comme indiqué ci-dessous après l'exécution de la commande suivante dans un terminal :

$ player simple.cfg

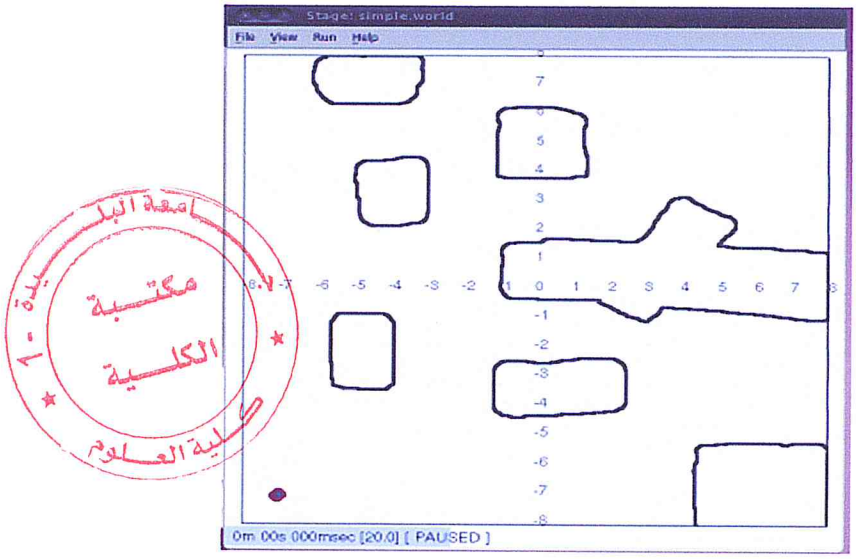


Figure 2 : feunetre de Stage

5. Avantage de Player/Stage

L'avantage de Player/Stage est la capacité de se déplacer à partir de la simulation d'un robot en changeant quelques paramètres. Ce passage permet une plus large gamme de tests qui peuvent être effectués avant de passer à un robot réel. Les connexions réseau entre les différents ordinateurs permettent le contrôle et la supervision des robots en temps réel à un terminal distant au lieu d'avoir à enregistrer puis contrôler les données.

6. Inconvénient de Player/Stage

L'inconvénient est le temps d'apprentissage de ce logiciel. En raison de l'indisponibilité d'un guide complet, il faut beaucoup de temps pour apprendre à utiliser Player/Stage donc ce n'est pas amical aux débutants. Un autre inconvénient du Player/Stage est le grand nombre de dépendances requises pour le faire fonctionner à son maximum.