REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'enseignement supérieur et de la recherche scientifique

Centre de développements et des technologies avancées

Université Saad Dahlab Blida

Département Informatique

Mémoire de Fin d'étude

Pour l'obtention du diplôme Master en Génie des Systèmes Informatique

Thème :

Planification de Trajectoires sans Collisions pour un Système Multi-Robots

---

Réalisé par :

- Mlle Rebbach Mouna

Proposé et dirigé par :

- Mr Maoudj Abderraouf
- Mme Mancer Yasmine

Soutenue publiquement le : 23/06/2016 devant le jury composé de :

- Mr M. Hamouda — Président
- Mme N. Toubaline — Examinatrice

2015/2016

« N'essayez pas de devenir un homme de succès.

Essayez de devenir un homme de valeurs. »

-Albert Enstein-

Résumé

Le sujet s'intéresse à la planification de trajectoires avec un évitement de collision dans les Système Multi Manipulateurs, opérant dans un espace de travail partagé.

L'approche proposée consiste à utiliser la négociation entre les différents agents constituant l'architecture de contrôle afin d'échanger les différentes positions (x, y, z), de chaque articulation, prises par le bras manipulant de chaque robot. Ces positions sont obtenues par un générateur flou de trajectoires, basé sur la logique floue. Elles seront utilisées pour la détermination des équations de droites, supposées constituant les différents segments des robots, afin de vérifier l'existence (ou non) de collision(s) entre ces droites (donc entre les robots). Finalement, l'approche proposée et développée sera intégrée dans notre Architecture Multi Agents de contrôle des Système Multi Robots.

Mots clés : Systèmes Multi Robots, Architecture Multi Agents de contrôle, Planification de trajectoires, Générateur flou de trajectoires, Évitement de collisions.

Abstract

In this work, we focus on the trajectory planning with collision avoidance in multi-manipulator systems, operating in a shared workspace.

The proposed approach is based on negotiation between the agents of the control architecture to exchange their joints positions (x, y, z) of each articulation, which are obtained by our trajectory pacificator that based on fuzzy logic. These will be, used to calculate the equations of lines, supposedly constituting the various segments of robots, to verify the existence (or not) of collision (s) between these lines (so between robots). Finally, the proposed approach will be integrated into our multi-agent architecture for control of multi-robot systems.

Keywords: Multi-robot systems, Multi-agent control architecture, Trajectory planning, Fuzzy generator trajectories, Collision avoidance.

ملخص

يهدف هذا الموضوع إلى تخطيط مسارات عدة روبوتات ذات ذراع مناور تتقاسم مهامها في نطاق عمل واحد مع اجتناب الاصطدام. حيث نعتمد في الإستراتجية المتبعة على تبادل الوضعيات بين الوكيل الممثل للربوت و الوكلاء الآخرين المشكلة للنظام. هذه الوضعيات التي يأخذها أثناء الحركة تقدم على شكل إحداثيات ديكارتية متحصل عليها من مولد ضبابي

المسارات يعتمد على المنطق الضبابي. أين تدخل في تحديد معادلة القطعة المستقيمة المفروض أنها تمثل رياضيا احد القطع المشكلة لذراع الربوت .فتستعمل لتأكد من وجود اصطدام أو غيابه بين اذرع الربوتات .في الأخير تدمج هذه الفكرة في بنية تحكم متعددة الوكلاء لنظام متعدد الربوتات.

الكلمات المفتاحية : نظام متعدد الربوتات . بنية تحكم متعددة الوكلاء . تخطيط المسارات . مولد ضبابي المسارات . اجتناب الاصطدام .

Remerciements

J'adresse en premier lieu ma reconnaissance à mon ***DIEU*** *le tout puissant, qui m'a donné la santé et la volante d'entamer et de terminer ce mémoire.*

J'adresse le grand remerciement à ma promotrice ***Mme Mancer Yasmine,***

pour ses conseils, son soutiens moral et ses dirigés du début à

la fin de ce travail.

Ce travail ne serait pas aussi riche et n'aurait pas pu avoir le jour sans l'aide

et l'encadrement de ***Mr Maoudj Abderraouf****, je le remercie pour la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur, son accueil dans son lieu de*

travail et sa disponibilité durant ma préparation de ce mémoire.

Je suis consciente de l'honneur que m'a fait

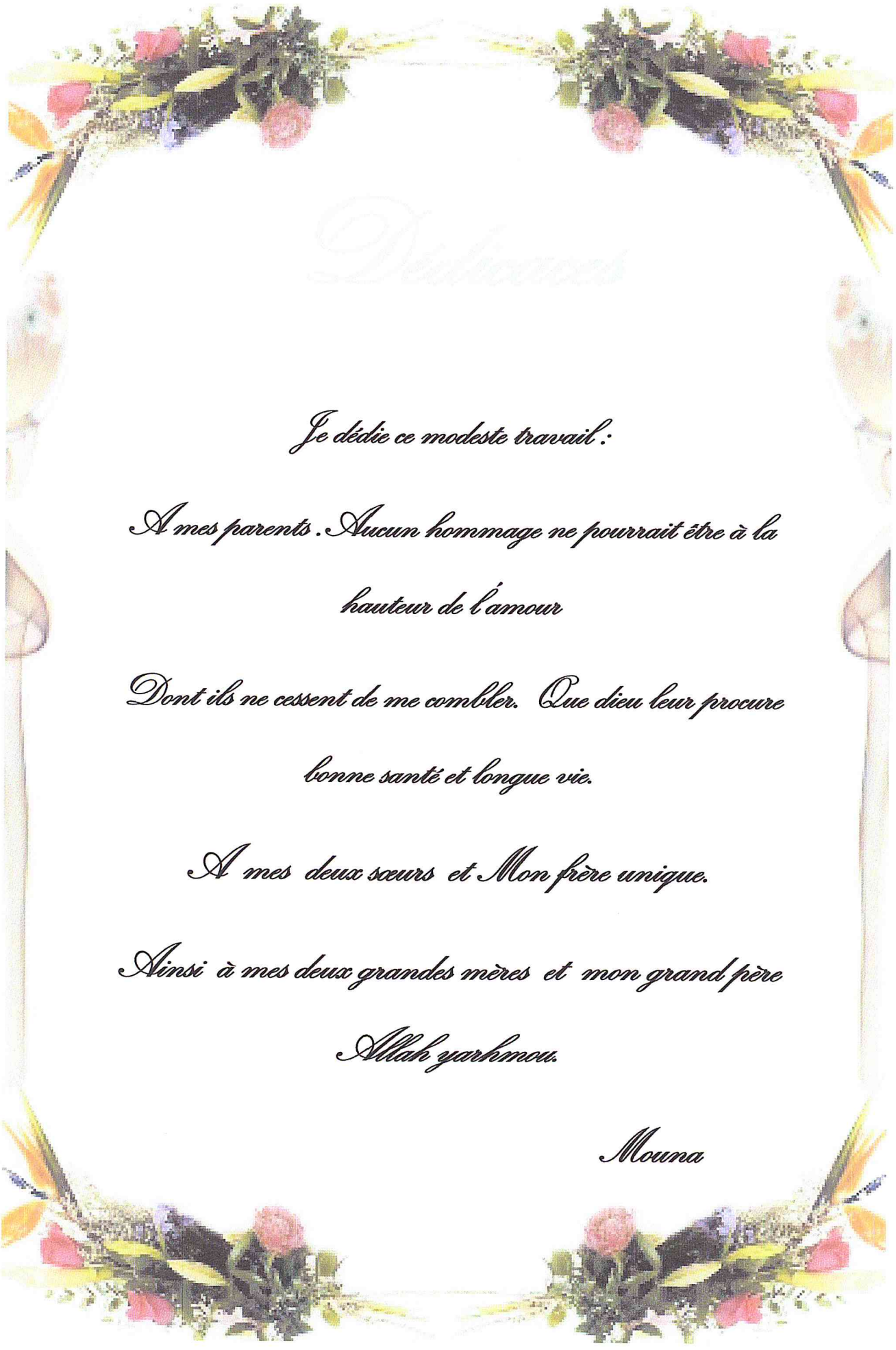
le membre du jury d'avoir accepté d'examiner mon travail.

Mes remerciements s'adressent également à tous mes professeurs pour leurs générosités et la grande patience dont ils ont su faire preuve malgré leurs charges académiques et professionnelles.

J' exprime mes profonds remerciements à l'encontre de mes ***parents****, mon frère et sa femme , et mes deux sœurs et leurs maris , qui m' ont enseigné la patience, le sacrifice,*

et qui ont toujours été là pour moi.

Je n'oublie pas de dire un grand merci à toutes les personnes, et amis, qui ont contribuées de près et de loin à l'enrichissement de mon travail et à mon épanouissement intellectuel.



Je dédie ce modeste travail :

A mes parents . Aucun hommage ne pourrait être à la hauteur de l'amour

Dont ils ne cessent de me combler. Que dieu leur procure bonne santé et longue vie.

A mes deux sœurs et Mon frère unique.

Ainsi à mes deux grandes mères et mon grand père

Allah yarhmou.

Mouna

Table des matières

Liste des Tables

Liste des figures

Liste des acronymes

DMACA : Distributed Multi-agent Control Architecture.

GFTs : Générateur Flou de Trajectoires.

IA : Intelligence Artificielle.

JADE : Java Agent Development Framework.

LA : Local Agent.

LF: Logique floue.

MGD : Modèle Géométrique directe.

MGI : Modèle Géométrique Inverse.

OT : Organe Terminal.

PTs: Planification de trajectoires.

RA :.Remote Agent

RM: Robot Manipulateur.

RMs: Robots Manipulateurs.

SA : Supervisory Agent.

SMA : Système Multi Agents.

SMM : Système Multi Manipulateurs.

SMR : Système Multi Robots.

6DDL: Six degrés de liberté.

Introduction générale

De l'appareil le plus simple aux machines humanoïdes, en passant par l'indexation de pages web, le terme robot regroupe beaucoup de notions. Il a été employé pour la première fois par le célèbre dramaturge tchèque Karel Capek (1890-1938) dans la pièce de théâtre R. U. R. (Rossum's Universal Robots) de 1921. Ce mot dérive du mot tchèque "robota" qui signifie la corvée. Bien que le premier robot manipulateur soit apparu en 1954 avec le dépôt d'un brevet par George Devol [33].

L'organisation internationale de normalisation (ISO) a officiellement défini un RM comme une machine automatiquement commandée, reprogrammable et possédant trois axes ou plus [33].

Pour chaque tâche spécifique comme par exemple la soudure, l'encollage, la peinture ou l'assemblage, des solutions ont été proposées. Les bras manipulateurs qui évoluent dans des environnements structurés et entièrement connus exécutent des séries de mouvements à grande vitesse préalablement appris. La présence des robots manipulateurs dans l'industrie est aujourd'hui banalisée notamment dans le domaine de l'automobile. Ces robots sont conçus pour fonctionner dans des environnements dont tous les paramètres sont précisément contrôlés [33].

Cependant, les SMM sont incapables de planifier leurs trajectoires sans reconfigurations ou reprogrammations. En outre, le contrôle devient difficile lors de leurs mouvements pour éviter les collisions qui peuvent exister entre les robots, s'ils opèrent dans un espace de travail réduit.

Dans ce contexte, nous avons proposé une approche qui sert à planifier les trajectoires sans collisions d'un RM de six degrés de liberté basée sur la LF. Le but est de satisfaire les contraintes introduites par son espace de travail, et de bien contrôler les mouvements au sien d'un tel SMR.

Les objectifs de notre approche sont :

- ✓ Trouver une solution qui permet aux RMs travaillant en coopération dans un espace partagé, de résoudre leurs problèmes de collisions s'ils existent par négociations.

- ✓ Si un des segments constituant le bras manipulant tombe en panne, le RM sera capable de trouver le chemin vers la cible par la solution trouvée.
- ✓ Implémenter un module capable de générer les trajectoires des RMs, basé sur la logique floue, avec un nombre de règles d'inférences réduit, sans utiliser une combinaison de solutions existantes pour la PTs.
- ✓ La solution implémentée doit permettre aux RMs de converger vers n'importe quelle position proche de son espace, et non pas prédéfinie (cas des trajectoires avec des positions fixes), où les positions constituant la trajectoire générée convergente vers la cible, sont à chaque fois changeables, permettant au robot de prendre plusieurs chemins vers la même cible.

Notre mémoire est organisé de la manière suivante :

- Le premier chapitre présente des généralités sur la robotique. Il définit les notions de bases, les domaines d'applications, ainsi qu'un descriptif général sur les SMR.
- Le deuxième chapitre donne des rappels sur les Systèmes Multi Agents, les différentes organisations existantes pour une flotte d'agents, et l'architecture de contrôle proposée.
- Le troisième chapitre expose en premier temps les notions de mécanique du mouvement des corps et notamment la modélisation géométrique des robots manipulateurs. Par la suite, il donne une étude comparative des approches existantes pour la planification de trajectoires.
- Le quatrième chapitre représente le cœur de notre sujet, il illustre les deux méthodes proposées pour la PTs. Une sert à créer un générateur flou basé sur la LF, qui fonctionne hors ligne, et l'autre méthode est une suite aux traitements obtenus par le GFTs, afin de contrôler l'exécution de la trajectoire générée, de telle sorte à éviter la présence d'une collision en mode en ligne entre les RMs.
- Le cinquième chapitre vient pour clôturer le travail de ce mémoire, par l'intégration des méthodes proposées dans le cadre de PTs sans collisions, pour un système multi robots manipulateurs. Il présente une conception détaillée de l'architecture générale du système, et des implémentations assurées par des tests expérimentaux qui sont discutés.

Chapitre I :

Généralités sur les robots

I.1. Introduction

L'utilisation des robots est aujourd'hui couramment envisagée pour l'automatisation de nombreuses tâches. Celles-ci sont particulièrement diversifiées: le nettoyage, le transport dans les ateliers automatisés, l'agriculture, l'exploitation des mines, l'assistance aux personnes handicapées et l'exploration de milieux hostiles en sont quelques exemples.

On va entamer ce chapitre par un aperçu historique sur les robots, leurs caractéristiques, et leurs domaines d'application, ainsi que les SMR.

I.2. Notions de base

Le monde des robots est très vaste, parmi ses éléments de base, on peut citer les suivants :

I.2.1. Domaine robotique

Ensemble des études et des techniques de conception et de mise en œuvre des robots effectuant des tâches déterminées en s'adaptant à leur environnement [1]. La robotique vient de l'anglais robotics, imaginé par le romancier « Isaac Asimov » et popularisé par un livre publié en 1942, (Runaround)[2].

I.2.2. Automate

Machine qui, par le moyen de dispositifs mécaniques, pneumatiques, hydrauliques, électriques ou électroniques, est capable d'actes imitant ceux des corps animés[3].Structure physique ou informationnelle qui fonctionne d'après des règles strictes, heuristiques ou

probabilistes, sans l'intervention consciente de l'être humain [4]. La Figure I.1 illustre un exemple d'automate.

Figure I.1. *Automate d'Hugo Cabret* [6].

I.2.3. Robot

Un robot est un automate doté de capteurs et d'effecteurs lui donnant une capacité d'adaptation et d´emplacement proche de l'autonomie. Un robot est un agent physique réalisant des tâches dans l'environnement dans lequel il évolue [3].

I.2.4. Intelligence artificielle

Discipline scientifique relative au traitement des connaissances et au raisonnement, dans le but de permettre à une machine d'exécuter des fonctions normalement associées à l'intelligence humaine : compréhension, raisonnement, dialogue, adaptation, apprentissage, ….etc [2].

I.3. Historique

On peut schématiquement distinguer trois principales ères en robotique : les automates, les robots n'étant pas dotés d'intelligence artificielle, et ceux disposant d'une IA.

I.3.1. Automates (1ère génération)

Un automate, contrairement à un robot (même s'il ne dispose pas d'IA), obéit uniquement à un programme préétabli, que ce soit de manière mécanique ou électronique. De ce fait, il n'y a aucune adaptation possible entre l'automate et son environnement.

On attribue la paternité du tout premier automate de l'humanité à Architos de Tarente (IVème siècle avant J.C). Il s'agissait d'une représentation d'un pigeon capable de voler et étant propulsé par de la vapeur. Malheureusement, aucun vestige ni aucun schéma ou représentation fidèle n'ont été retrouvés.

Les toutes premières traces d'automates remontent à l'antiquité par Héron d'Alexandrie au Ier siècle après J.C., ses réalisations ornèrent les temples et les théâtres de la ville égyptienne.

En 1495 Léonard de Vinci présenta un chevalier humanoïde capable de s'asseoir, de relever sa visière et de bouger ses bras.

L'automate le plus célèbre est le canard mécanique de Jacques De Vaucanson, capable d'ingurgiter de la nourriture et de la digérer tout en se déplaçant. De Vaucanson aurait également présenté en 1738 un second automate représentant un homme jouant d'un instrument à vent.

I.3.2.Robots (2ème génération)

Les robots de 2ème génération disposent d'organe(s) sensoriel(s), autrement dit des capteurs, pouvant influer sur leurs comportements. Ils sont donc relativement adaptables à leur environnement.

Le chien électrique de Hammond et Miessner (1915) est le premier robot de ce genre. Il se déplace selon la luminosité de l'endroit grâce à son capteur optique.

Le chien Phillidog de Henri Piraux en 1928, et le renard de Ducrocq (1953) fonctionnent selon le même principe.

Walter Grey équipe en 1950 sa tortue cybernétique de capteurs tactiles et lumineux.

I.3.3. Robots dotés d'IA (3ème génération)

Pour être intégré dans cette famille, un robot doit parvenir à effectuer une tâche par lui-même, sans aucune aide extérieure.

En 1973, l'université de Waseda présente le tout premier humanoïde "intelligent" dénommé Wabot-1.Il est doué de la vision, peut manipuler des objets, effectue un semblant de marche et est même capable de débuter une conversation en japonais.

Hi-T-Hand d'Hitachi en 1974 manipule des aiguilles à travers des trous grâce à la détection de force.

Vers la fin des années 70 Hans Morava présente les premiers robots capables d'évoluer à l'extérieur, de façon autonome [5].

I.4. Classification des robots

On peut les classer en trois catégories [32]:

I.4.1.Manipulateurs

Ce sont des robots industriels destinés à effectuer des taches bien déterminées et de manière répétitive. A noter que pour ce type de robot :

- Les trajectoires sont bien déterminées dans l'espace,
- Les positions sont discrètes avec 2 ou 3 valeurs par axe,
- La commande est séquentielle.

La figure suivante illustre les différents éléments qui constituent un robot manipulateur :

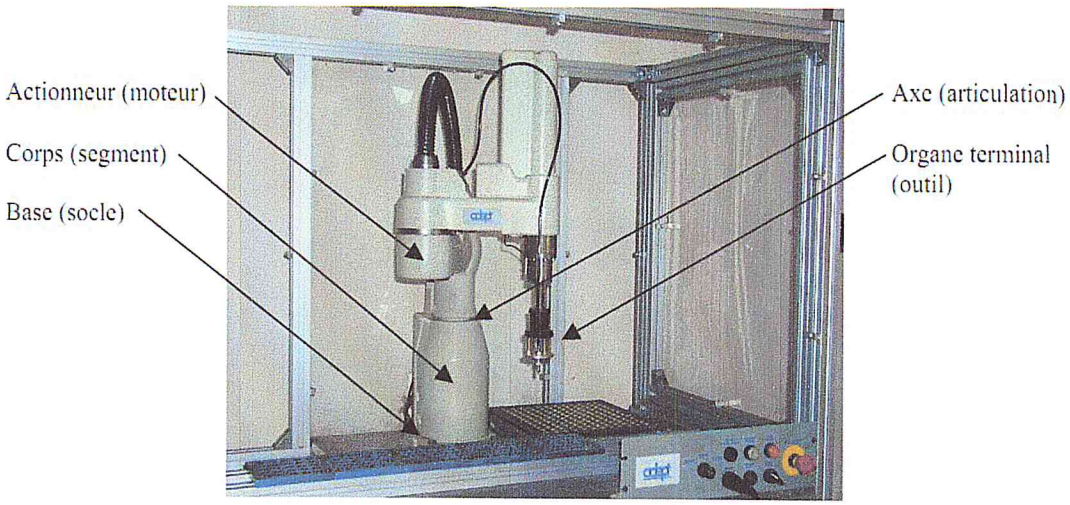


Figure I.2. *Constitution d'un robot manipulateur* [32].

Sous le terme organe terminal, on regroupe tout dispositif destiné à manipuler des objets (dispositifs de serrage, dispositifs magnétiques, à dépression, …), ou à les transformer (outils, torche de soudage, pistolet de peinture, …). En d'autres termes, il s'agit d'une interface permettant au robot d'interagir avec son environnement.

Un OT peut être multifonctionnel, au sens où il peut être équipé de plusieurs dispositifs ayant des fonctionnalités différentes. Il peut aussi être monofonctionnel, mais interchangeable. Un robot, enfin, peut-être multi-bras, chacun des bras portant un OT différent.

Le système mécanique articulé (S.M.A.) est un mécanisme ayant une structure plus ou moins proche de celle du bras humain. Il permet de remplacer, ou de prolonger, son action. Son rôle est d'amener OT dans une situation (position et orientation) donnée, selon des caractéristiques de vitesse et d'accélération données.

Précisons la notion d'articulation : Une articulation lie deux corps successifs en limitant le nombre de degré de liberté de l'un par rapport à l'autre.

Ces différents éléments du RM peuvent être regroupés en quatre parties principales comme indiqué sur la (figure I.3).

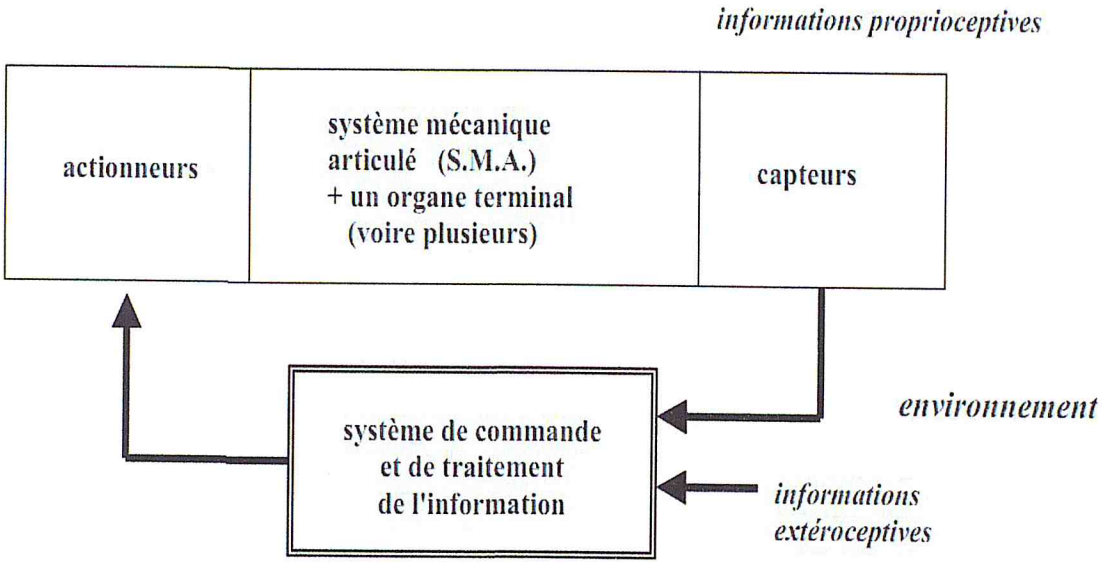


Figure I.3. *Parties constituant un RM* [32].

I.4.2. Télémanipulateurs

Appareils de manipulation à distance (pelle mécanique, pont roulant,…), apparus vers 1945 aux USA. Dans ce cas :

- Les trajectoires peuvent être quelconques dans l'espace,
- Les trajectoires sont définies de manière instantanée par l'opérateur.

I.4.3. Robots mobiles

D'une manière générale, on regroupe sous l'appellation robots mobiles l'ensemble des robots à base mobile, par opposition notamment aux RMs. L'usage veut néanmoins que l'on désigne le plus souvent par ce terme les robots mobiles à roues.

Les autres robots mobiles sont en effet le plus souvent désignés par leur type de locomotion, qu'ils soient marcheurs, sous-marins ou aériens. L'architecture mécanique d'un robot mobile dépend très largement des spécificités de la mission à effectuer et de l'environnement de travail. Ces données conditionnent entre autres le choix d'un système de locomotion approprié.

I.5. Caractéristiques d'un robot

Un robot doit être choisi en fonction de l'application qu'on lui réserve. Voici quelques paramètres à prendre en compte [32]:

- La charge maximale transportable (de quelques kilos à quelques tonnes), à déterminer dans les conditions les plus défavorables.
- L'architecture du Système Mécanique Articulé, le choix est guidé par la tâche à réaliser.
- Le volume de travail, définit comme l'ensemble des points atteignables par OT. Tous les mouvements ne sont pas possibles en tout point du volume de travail.
- Le type du robot et la mission à effectuer.
- La vitesse de déplacement.
- La masse du robot.
- Le coût du robot.

I.6. Domaines d'applications

La robotique est un domaine en plein essor depuis quelques années, utilisée dans des domaines extrêmement rigoureux et exigeants. Nous allons explorer ces différents domaines.

I.6.1. Domaine spatial

Les manipulateurs mobiles spatiaux sont destinés à explorer des environnements où l'homme ne peut pas se rendre, c.-à-d. des environnements souvent mortels pour l'homme [7]:

Aujourd'hui, l'histoire de la conquête spatiale est devenue indissociable de celle de la robotique et, à ce moment même, plusieurs robots sont en activité, aussi bien sur la Station Spatiale Internationale que sur la planète Mars. Les futures missions d'exploration feront elles aussi appel aux robots et à leur IA (figure I.4), que ce soit pour relever de nouveaux défis ou pour mieux connaître notre système solaire. Mais ces machines s'avèreront être de plus en plus variées puisque toutes les tailles, toutes les fonctions et tous les modes de déplacements sont à l'étude afin de développer des robots capables d'assurer des tâches très distinctes, de façon autonome ou en parfaite synergie avec les humains.

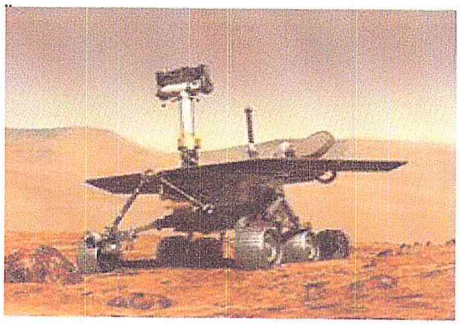


Figure I.4. *Robot Rover Martien* [8].

I.6.2. Agriculture

Heureusement, après des décennies d'expérimentation et de tâtonnements, les robots ont enfin fait leur entrée à la ferme. Cette machine totalement autonome fonctionne grâce à l'énergie solaire et circule dans les rangées de plantations pour surveiller et analyser les plants (voir figure I.5). Ce robot a déjà passé avec succès de nombreux tests réalisés dans des champs de légumes mais se contente de surveiller la « bonne santé » des cultures et plantations. Grâce à ses nombreux capteurs, senseurs et caméras, il détecte rapidement d'éventuelles anomalies (présence de mauvaises herbes, animaux nuisibles, croissance trop faible) et avertit l'exploitant agricole qui peut ainsi prendre immédiatement les mesures appropriées[11].

Figure I.5. *Robot utilisé en agricole* [12].

I.6.3. Domaine de service

La démocratisation de la robotique a conduit, ces dernières années, à voir de nombreux robots s'installer chez les particuliers pour effectuer des tâches à la place de leur possesseur, la figure I.6 illustre le robot de service ASIMO. En effet, ceux-ci sont capables de faire le ménage, tondre la pelouse, nettoyer la piscine... Ce qui conduit certains clients (aisés) à se procurer ces domestiques contemporains.

Enfin, la robotique, autrefois réservée à des applications précises ou coûteuses, est aujourd'hui de plus en plus utilisée à titre ludique. En effet, les robots compagnons par exemple sont des objets de plus en plus convoités : les applications « basiques » de jouet pour enfant, jusqu'à l'humanoïde destiné à remplacer une présence humaine [8].

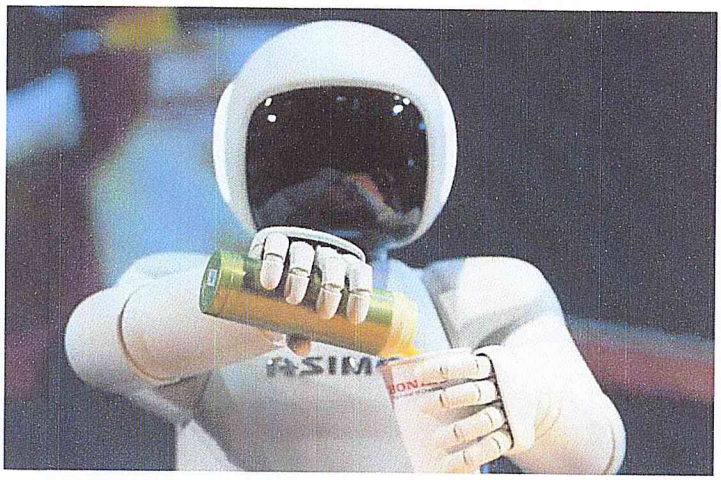


Figure I.6 : *Robot de service ASIMO* [10].

I.6.4. Domaine militaire

Les robots sont de plus en plus utilisés dans le domaine militaire. En effet, la miniaturisation permet aujourd'hui de créer des robots discrets mais dotés de nombreux capteurs, ce qui est idéal pour des missions d'espionnage ou d'éclairement, comme le montre la figure I.7 suivante.

De plus, certains robots sont équipés d'un armement pour évoluer en milieu hostile, dans le but de remplacer les soldats pour limiter les pertes humaines [8].

Figure I.7. *Robot utilisé dans le domaine militaire* [13].

I.6.5. Domaine médical

Les robots commencent à être de plus en plus dans le domaine médical, qu'il s'agisse de « simples » échographies ou d'opérations chirurgicales plus délicates. En fait ces robots ne sont pas complètement autonomes mais ils assistent les médecins ou chirurgiens, jusqu'à permettre des opérations médicales à distance (télémédecine). On parle de surgétique (mot né de l'anglais « surgery » : chirurgie) c'est-à-dire tout ce qui consiste à introduire les derniers outils des technologies informatiques et robotiques dans la pratique médico-chirurgicale. Cette pratique de « chirurgie assistée » est émergente donc bien que peu répandue, elle est en phase de devenir la chirurgie du futur [8]. La figure I.8, montre une opération à l'aide d'un robot.

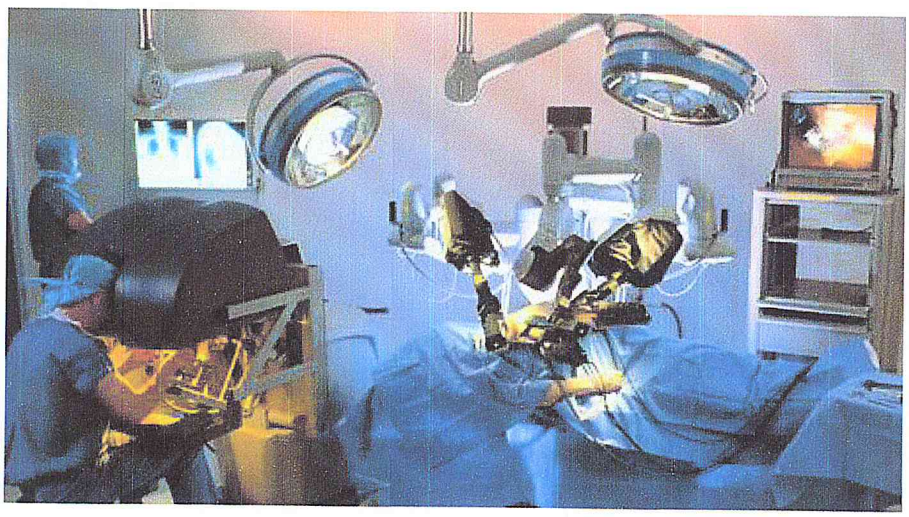


Figure I.8. *Endoscopie (Innovations en chirurgie cardiaque)* [9].

I.6.6. Domaine industriel

Depuis leur apparition sur les chaînes de production dans les années 1970, les robots industriels ont pris une place de plus en plus importante dans la plupart des secteurs où les opérations d'assemblage et les manipulations de produits sont récurrentes. Durant les trois dernières décennies, les progrès technologiques ont participé à rendre les robots plus flexibles, plus rapides et plus précis [14].

Aujourd'hui, ils jouent un rôle prépondérant pour :

- les manipulations de produits à haute cadence,
- les opérations de packaging avec des manipulations de produits (manipulations primaires) et d'emballage (manipulations secondaires),
- l'assemblage de produits (secteurs automobile, pharmaceutique, cosmétique, etc.) ci dessous (figure I.9).
- les opérations de fin de ligne (palettisation).

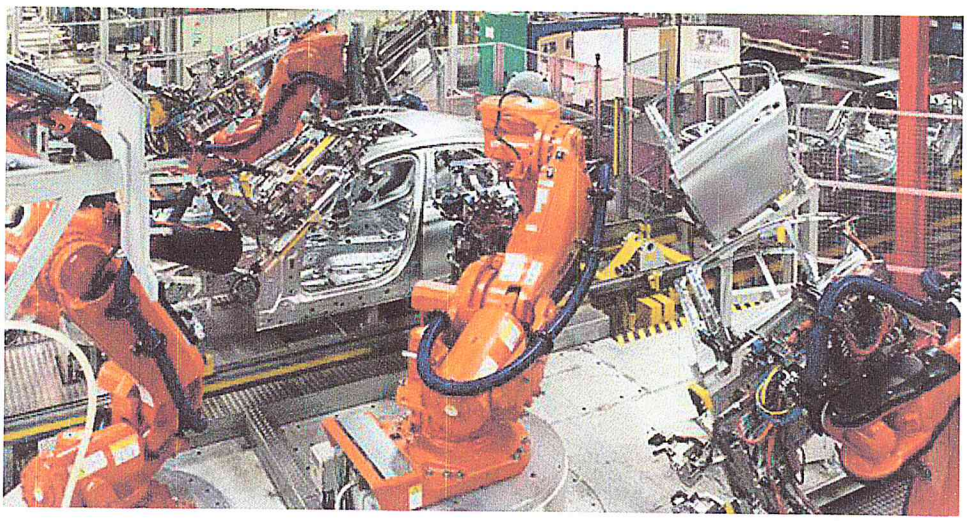


Figure I.9 : *Robots soudeurs et manipulateurs en action* [15].

D'après les domaines d'applications vus précédemment, on constate bien qu'un robot puisse avoir un rendement élevé, et des missions bénéfiques fournis pour son environnement par l'évolution des technologies robotiques. Mais, certaines tâches sont complexes, voire impossibles pour être effectuées comme des tâches séparées dans l'espace comme le cas d'une zone industrielle robotisée, qui planifie ses travaux et renforce sa production à l'aide

d'une flotte de robots ou bien dit un SMR qui sera présenté par la suite. Donc en robotique le fait d'utiliser une équipe de robots, de leur inculquer certains concepts sociaux, et non seulement d'utiliser un seul robot individuel peut avoir un impact majeur sur les capacités d'offres par les systèmes robotisés.

I.7.Systèmes Multi robots

La notion de systèmes multi-robots ou Multi-Robot Systems en anglais débute dans les années 1990, notamment dans des travaux regroupant des robots mobiles rassembleurs d'objets, des colonies de robots marcheurs [22].

Dans cette section on va faire un tour d'horizon sur ce type de système, en présentant dans la suite quelques notions liées à ce dernier.

I.7.1. Définition

C'est une flotte de robots autonomes constituée de plusieurs robots capables de partager des données et d'effectuer ensemble une ou plusieurs tâches. Ces systèmes sont de plus en plus envisagés en connexion et en coopération avec d'autres objets / capteurs fixes et mobiles dans l'environnement [21].

I.7.2.Coopératifs vs compétitifs

La notion de comportement collectif dans les SMR se définit par similitude avec le principe de travail collectif dans notre société. Popenoe [40] a défini un comportement collectif comme un comportement dans lequel il apparaît une réponse à une influence ou à un stimulus commun dans une situation spontanée, imprédictible, non-structurée et instable. Le comportement collectif comprend deux sous-catégories de comportements associées aux deux notions de coopération (appelé comportement coopératif) et de compétition (appelé comportement compétitif). Les environnements multi-robots sont ainsi soient coopératifs, soient compétitifs.

Une première définition présente la coopération par l'identification de situations dans lesquelles il apparaît un besoin d'interaction entre plusieurs robots. Cette coopération devient utile quand le résultat de l'exécution de la tâche est estimé meilleur avec plusieurs robots. En effet, la coopération est l'interaction entre des robots qui travaillent dans un intérêt commun.

La notion de comportements compétitifs présente les situations et les problèmes à résoudre sous la condition que les robots soient en compétition les uns contre les autres pour

satisfaire leur propre intérêt. Chaque robot possède dans ce cas, un ensemble d'intérêts partiellement ou totalement en concurrence avec les intérêts d'autres robots. Cette notion d'intérêt est définie par des fonctions d'utilité en accord ou en contradiction avec les autres robots [22].

I.7.3.Problèmes inhérents

Ce type de système ou bien dit une flotte de robots, peut engendrer plusieurs problèmes au niveau de l'environnement du travail commun partagé par les différents robots en coopération et en compétition cités ci-dessous [22] :

- Si plusieurs demandes visant la même ressource apparaissent simultanément, il se produit un conflit de ressources. Ce problème a été étudié sous de nombreuses formes : un cas bien connu de problème de ce type est le problème de l'exclusion mutuelle.
- Les problèmes de partage de média de communication sont souvent associés à des limitations de bande passante.
- La question de l'identification des sous-buts et de leur répartition dans un SMR.
- Le problème du partage de l'espace a été étudié principalement dans les problèmes de planification de mouvements multi-robots et dans les problèmes d'évitement de collision, de congestion et d'interblocage qui vont cerner par la suite la problématique de ce travail.

I.7.4. Coordination : statique vs dynamique

La coordination est une tâche essentielle dans les SMR. La performance globale du système est directement influencée par la qualité de la coordination et la qualité du contrôle de l'exécution de chaque mouvement de chaque robot impliqué. Cette coordination peut être statique ou dynamique. La coordination statique (également connue sous le nom de coordination délibérative, ou coordination hors-ligne), se réfère à l'adoption d'une convention entre plusieurs robots avant de s'engager dans une tâche. Dans certaines règles comme "garder la droite", "s'arrêter aux intersections" et "garder suffisamment d'espace devant soi" sont utilisées pour résoudre les problèmes de contrôle de trafic. La coordination dynamique (également connue sous le nom de coordination réactive, ou coordination en ligne) est faite

lors de l'exécution d'une tâche et basée sur l'analyse et sur la synthèse de l'information. Les informations sont obtenues par des moyens de communication inter-robots [22].

I.7.5. Communication

La communication, comme moyen de coordination, apparaît souvent comme un comportement rationnel dans les environnements multi-robots. Dans les faits, la communication est un mode d'interaction entre les robots. Par cette interaction, les robots peuvent d'une part partager les informations de position, l'état de l'environnement et les données des capteurs avec les autres dans le système.

D'autre part, chaque robot peut individuellement obtenir des informations sur les intentions, les objectifs et les actions de chacun des autres robots. Cao et al [41] ont classé la structure de communication en trois types selon le mode d'interaction, qui comprend : l'interaction via l'environnement, l'interaction via les perceptions et l'interaction via la communication.

I.7.6. Planification

Le principe de prévoir une séquence d'actions permettant d'atteindre un objectif est appelé planification. D'un point de vue opérationnel, il s'agit d'exécuter ce plan définissant une succession d'actions pour atteindre l'objectif recherché.

Dans un SMR, la planification est utilisée pour coordonner des robots leur permettant ainsi d'accomplir une mission. Dans le cadre des SMR, la planification optimale est un problème NP-difficile [22].

- **Planification de mouvements**

Le problème de la planification de mouvements, correspond à la question de la génération d'une série de mouvements continus d'une configuration initiale à une configuration finale dans l'espace des configurations (i.e. l'espace des états du robot défini par l'emplacement, l'orientation et les angles des articulations), tout en évitant les collisions avec les obstacles. Le mouvement est représenté comme un trajet dans l'espace de

configuration. La planification de mouvements est éminemment nécessaire pour un robot, puisque par définition, un robot accomplit des tâches en se déplaçant dans le monde réel.

La planification de mouvements multi-robots tient compte non seulement des obstacles (statiques ou dynamiques) de l'environnement, mais aussi des interférences possibles entre les robots. Répartis en équipe et destinés à effectuer des tâches indépendantes dans un espace de travail partagé, les robots deviennent mutuellement des obstacles mobiles pour chacun. Ainsi, chaque robot doit prendre en compte le mouvement des autres robots pour mener à bien sa mission [22].

- **Planification de tâches**

La planification de tâches multi-robots comprend deux principaux problèmes : la décomposition en tâches et la répartition de ces tâches vers un ensemble de robots. Les résultats de la répartition des tâches sont directement dépendants de la première étape de décomposition.

La décomposition en tâches multi-robots se réfère à la décomposition de la globalité de la mission ou à la décomposition d'une tâche de cette mission. Cette mission ou cette tâche est décomposée en plusieurs sous-tâches simples qui peuvent être réalisées indépendamment par un ou plusieurs robots, selon la caractéristique de la mission, l'exigence recherchée et la répartition des ressources de la mission [22].

I.8. Conclusion

La robotique est un très bon exemple du domaine pluridisciplinaire qui implique de nombreuses thématiques telles que la mécanique, la mécatronique, l'électronique, l'automatique, l'informatique et l'intelligence artificielle.

L'expansion des robots dans les différents domines est une excellente chose, car elle permet d'accomplir des tâches dangereuses et monotones pour l'Homme et leur rendement est beaucoup plus avantageux et stimulant pour l'économie.

Dans ce chapitre, on a abordé quelques notions de base sur la robotique, et notamment sur les RMs qui vont être utilisés par la suite pour notre SMR. Le chapitre suivant sera consacré aux SMA.

Chapitre II :

Architecture de contrôle multi-agents proposée pour SMR

II.1. Introduction

Les travaux étudiés dans ce chapitre se concentrent sur les SMA, où un robot est considéré comme étant un agent, autonome et physiquement indépendant. Notre thème d'investigation est limité aux systèmes multi-robots. Aussi, nous allons décrire dans ce chapitre l'organisation ou l'architecture multi-agents proposée pour le contrôle d'un système robotique constitué de RMs. Cette architecture doit être basée sur un modèle générique pour permettre l'ajout de nouvelles fonctionnalités et de nouveaux équipements.

II.2. Système multi-agents

Un SMA est un ensemble organisé d'agents, constitué d'une ou plusieurs organisations qui structurent les règles de cohabitation et de travail collectif entre agents.

II.2.1. Définition d'un agent

Un agent peut être défini comme une entité (physique ou abstraite) capable d'agir sur elle-même et son environnement, disposant d'une représentation partielle de cet environnement, pouvant communiquer avec d'autres agents et dont le comportement est la conséquence de ses observations, de sa connaissance et des interactions avec les autres agents [23]. La figure II.1 ci-dessous présente la structure générale d'un agent.

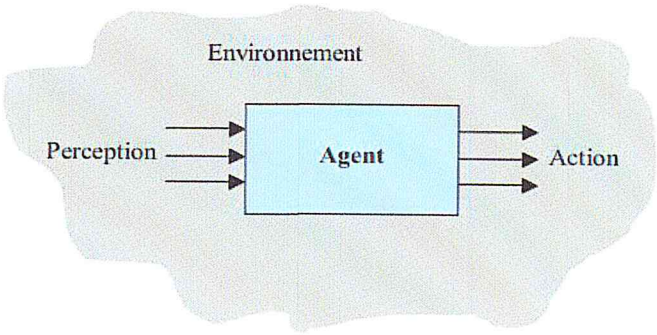


Figure II.1. *Aperçu externe et général d'un agent* [24].

II.2.2. Caractéristiques d'un agent

La définition précédente ressort les cinq principales propriétés qui caractérisent un agent [39] :

- **Autonome :** l'agent peut spontanément effectuer certaines tâches et peut prendre des initiatives.
- **Communicatif/social :** Un agent devrait avoir un niveau élevé de communication avec d'autres agents.
- **Réactif :** Un agent devrait pouvoir percevoir son environnement (qui peut être un monde physique, un utilisateur via une interface graphique ou une collection d'autres agents) et réagir à ses changements, que ce soit la modification des objectifs de l'utilisateur ou des ressources disponibles.
- **Proactif :** les agents n'agissent pas simplement en réponse à leur environnement, ils sont capables d'exposer un comportement dirigé but en prenant l'initiative.
- **Adaptatif :** un agent adaptatif est un agent capable de contrôler et d'adapter ses aptitudes (communicationnelles, comportementales, etc.) en réponse aux connaissances internes ou aux changements de l'environnement.
- **Orienté-but/intentions :** ces agents ont un plan d'action interne explicite pour accomplir un but ou un ensemble d'objectifs.
- **Persistance :** les agents persistants ont un état interne qui reste cohérent.
- **Mobilité :** les agents mobiles peuvent décider d'émigrer à une machine différente ou à un autre réseau tout en maintenant la persistance.
- **Emotion :** agents avec la capacité d'exprimer l'émotion ou l'humeur comme un être humain. De tels agents pourraient également avoir une certaine forme de caractère ou d'aspect anthropomorphe.
- **L'intelligence :** agents avec la capacité de raisonner. Un agent est intelligent s'il est capable de réaliser des actions flexibles et autonomes pour atteindre les objectifs qui lui ont été fixés. La flexibilité signifie la réactivité, la proactivité et les attitudes sociales.

Les agents ont donc deux tendances. une première tendance sociale tournée vers la collectivité (mécanismes et connaissances associées concernant les mécanismes du groupe). Une deuxième tendance individuelle avec des mécanismes et des connaissances contenant les règles de fonctionnement interne de l'agent.

II.2.3. Différents types d'agents

La granularité des agents impliqués dans une application varie selon deux écoles coexistant aujourd'hui : l'école cognitive et l'école réactive. Suivant le type d'agent utilisé, on parlera de systèmes cognitifs ou de systèmes réactifs.

- **Agent cognitifs :**

Les agents cognitifs disposent d'une base de connaissances comprenant les diverses informations liées à leur domaines d'expertise et à la gestion des interactions avec les autres agents et leur environnement. Les agents sont généralement « intentionnels » c'est-à-dire qu'ils possèdent des buts et des plans explicites leur permettant d'accomplir leurs buts.

- Avantage : comportement guidé par des intentions.
- Inconvénient : manque de réactivité, coût de décision élevé.

- **Agents réactifs :**

Les agents réactifs au contraire ne sont pas « intelligents» pris individuellement. Ils peuvent que réagir à des stimuli simples provenant de leur environnement, et leur comportement est alors simplement dicté par leur relation à leur entourage sans que ces agents ne disposent d'une représentation des autres agents ou de l'environnement.

- Avantages : forte réactivité au contexte, faible coût de décision.
- Inconvénient : manque de cohérence du comportement global.

Agent cognitif	*Agent réactif*
Représentation explicite de l'environnement	Pas de représentation explicite
Peut tenir compte de son passé	Pas de mémoire de son historique
Agents complexes	Fonctionnement stimulus/réponse
Système composé de petit nombre d'agents	Système composé de grand nombre d'agents

Table II.1. *Tableau comparatif pour les agents cognitifs et réactifs* [24].

- **Agent hybride :**

En général, la différence entre des agents réactifs et des agents cognitifs peut être expliquée par le compromis efficacité/complexité. La complexité des systèmes réactifs exige le développement de nouvelles théories dans le domaine de la coopération, de la communication et de la compréhension de nouveaux phénomènes tels que l'émergence.

Toutefois, il est maintenant possible de concevoir des systèmes hétérogènes comportant les deux types de comportement (cognitif et réactif) : on parlera alors d'agents hybrides.

II.2.4. Types d'organisation

Dans la littérature, on distingue plusieurs types d'organisations pour les SMA [28]:

II.2.4.1. Approche centralisée

Il s'agit de l'approche la plus classique et la plus ancienne, la figure II.2 présente son architecture. Elle se caractérise par une prise de décision localisée au sein d'un agent unique qui supervise l'exécution de tout le système. Il gère, aussi, en temps réel, les événements qui surviennent tout au long de l'exécution du système composé de plusieurs agents .

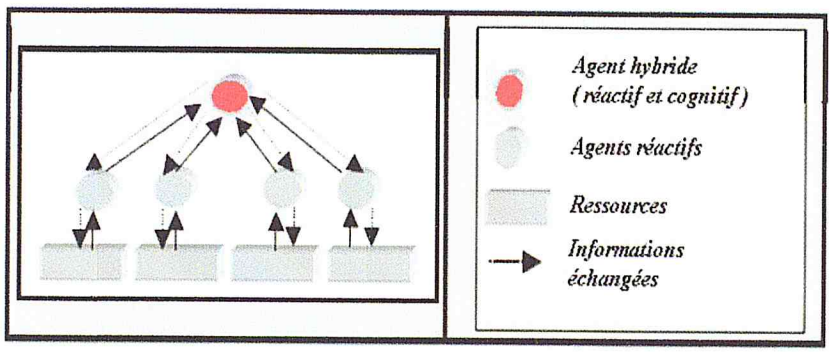


Figure II.2. *Approche centralisée* [28].

II.2.4.2. Approche hiérarchisée

Dans ce type d'approche illustré par la figure II.3, chaque agent d'un niveau donné coordonne les agents du niveau inférieur, et ce jusqu'au niveau le plus bas. Donc, chaque niveau a des relations de dépendance du niveau supérieur et de dominance du niveau inférieur. Chaque décision est élaborée au niveau où un problème est détecté. Les agents des

niveaux inférieurs traitent cette décision comme une contrainte et transmettent en retour une information de suivi à l'agent du niveau supérieur. Ainsi, cette alternative basée sur une décomposition du problème global en sous-problèmes, et sur le concept d'agrégation, est souvent utilisée. Le principe de décomposition permet de ramener la résolution du problème global à la résolution successive de sous-problèmes de dimension et de complexité acceptables.

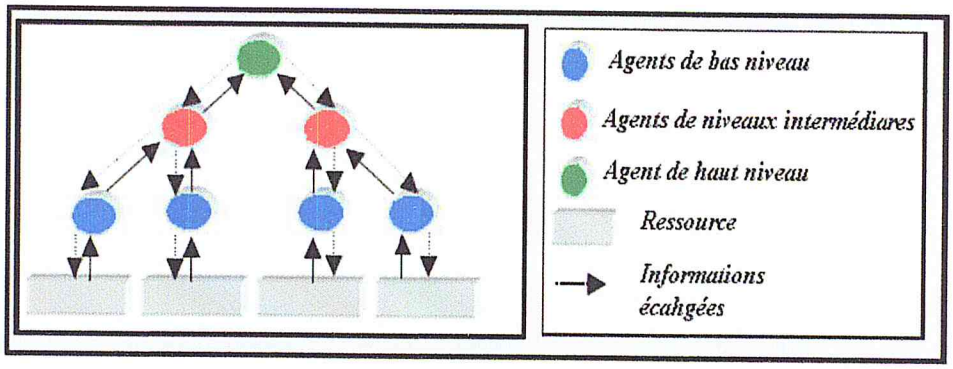


Figure II.3. *Approche hiérarchisée distribuée supervisée* [28].

II.2.4.3. Approche coordonnée

L'approche coordonnée correspond à un ensemble de structures hiérarchisées (voir la figure II.4). La différence est l'existence d'une coopération entre les agents d'un même niveau. Ces structures accroissent théoriquement la capacité de décision au sein de chacun de ces niveaux. Ainsi, cette coopération doit améliorer la réactivité en proposant une décomposition dynamique du problème au niveau où il apparaît, plutôt qu'une décomposition hiérarchique des ordres vers les niveaux inférieurs.

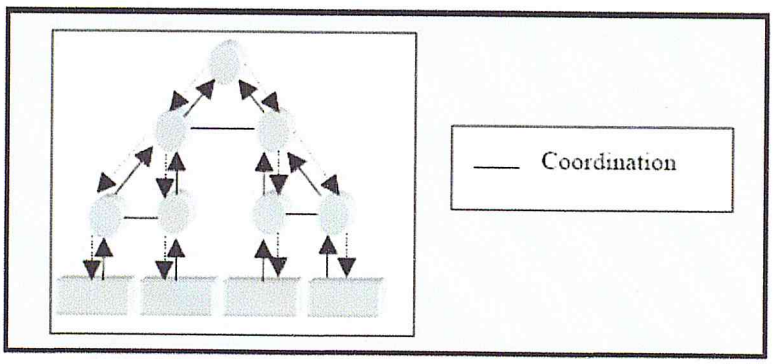


Figure II.4. *Approche coordonnée* [28].

II.2.4.4. Approche distribuée

Elle est fondée sur une distribution complète de la décision sur l'ensemble des agents. Le contrôle d'une telle structure est beaucoup plus complexe en raison de sa modularité, par rapport à une structure hiérarchique qui est plus rigide mais plus facilement maîtrisable. Cette structure présente un niveau élevé de réactivité et de flexibilité. La figure II.5 suivante présente un modèle d'une organisation distribuée.

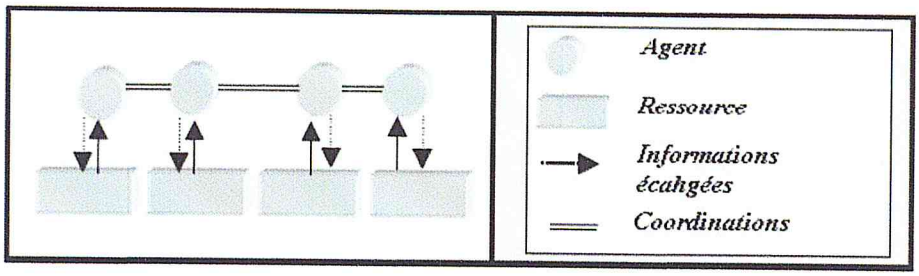


Figure II.5. *Approche distribuée* [28].

II.2.4.5. Approche distribuée supervisée

Une organisation distribuée supervisée se caractérise par un ensemble d'agents coopérants sous le contrôle d'un seul agent superviseur (voir figure II.6) dont le rôle est d'imposer, de conseiller ou de modifier une décision afin de respecter un objectif global. Un superviseur, qui possède une vision globale du système est rajouté à l'approche distribuée. Cette approche est un compromis entre les structures distribuée et centralisée. Elle permet une gestion globale et efficace par la centralisation de contrôle, d'une part, et une meilleure réaction aux perturbations par la distribution des capacités de décision d'autre part.

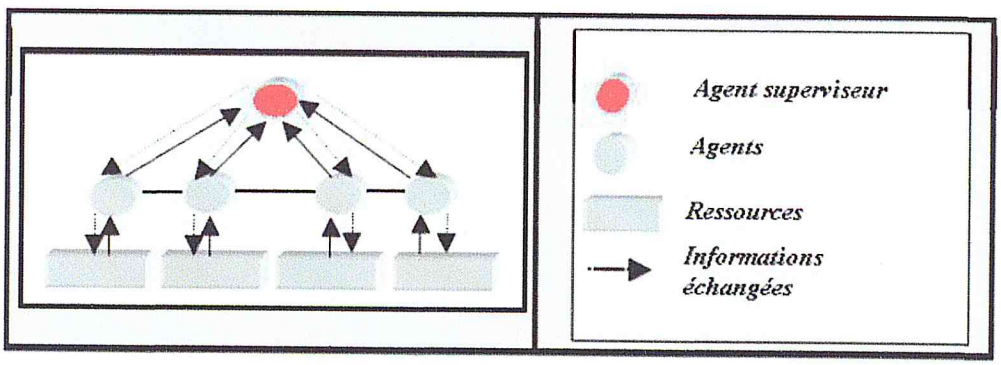


Figure II.6. *Approche distribuée supervisée* [28].

Architecture de contrôle multi-agents proposée pour le SMR

Le schéma suivant illustre l'architecture globale d'une flotte d'agents constituant un SMA, présentée par la figure II.7.

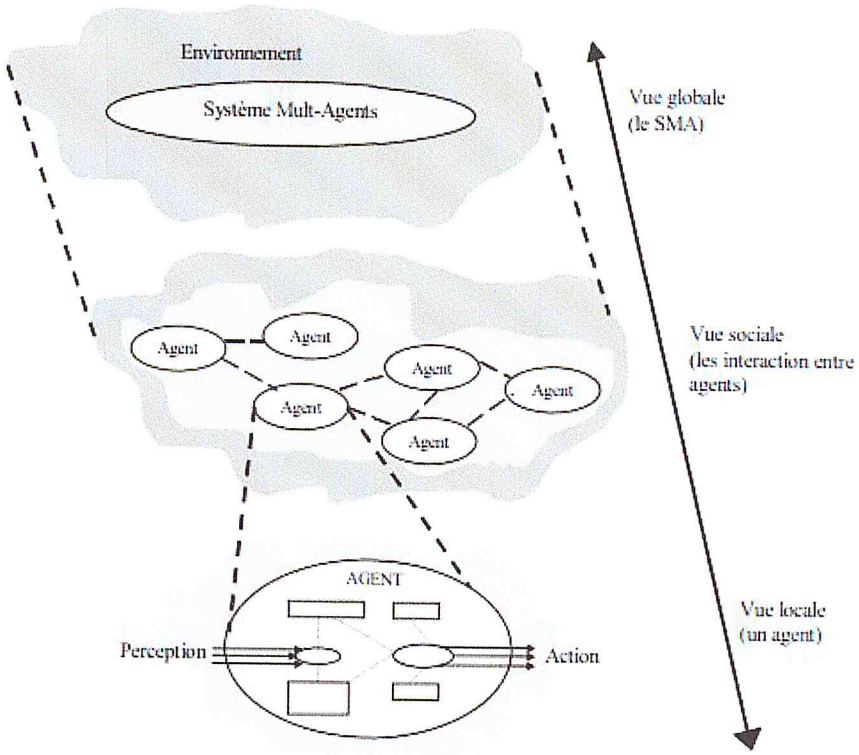


Figure II.7. *Système Multi-Agents vu selon différents niveaux de détail* [24].

II.3.Architecture de contrôle multi-agents proposée

Pour des raisons de modularité et flexibilité, une approche distribuée basée sur le SMA a été utilisée. Les SMA assurent à la fois la distribution de la décision, l'autonomie, la réactivité et la tolérance aux pannes, ainsi que la coordination des actions dans le cas des conflits. On va revoir son intérêt dans la communication entre agents pour résoudre le problème de collisions.

Afin de contrôler et superviser au mieux notre SMR, une architecture multi-agents de contrôle a été implémentée. La figure II.8 montre l'architecture multi-agents proposée dans [29]. Cette architecture est appelée Distributed Multi-agent Control Architecture. Elle est répartie sur deux côtés (côté maître et côté esclave) reliés par un réseau, dans lequel trois types d'agents autonomes Supervisory Agent, Local Agent et Remote Agent ont été définis. Les deux derniers types d'agents sont définis pour chaque robot dans notre système [29].

Architecture de contrôle multi-agents proposée pour le SMR

Le choix de ce modèle de SMA est justifié, d'une part, par le caractère générique du modèle de l'agent proposé (indépendant de la nature du robot à contrôler) et, d'autre part, par la possibilité d'intégration de l'ensemble dans un système robotique distribué (une cellule flexible de production par exemple) [30].

Le contrôle multi-agents est souhaitable (i) quand plus d'un robot est utilisé pour effectuer des tâches, (ii) quand un ensemble de robots fonctionnent indépendamment sur des tâches multiples dans un environnement partagé, ou (iii) quand un seul robot doit coordonner ses propres ressources par exemple pour un manipulateur mobile, coordonner le manipulateur et la base mobile) [31].

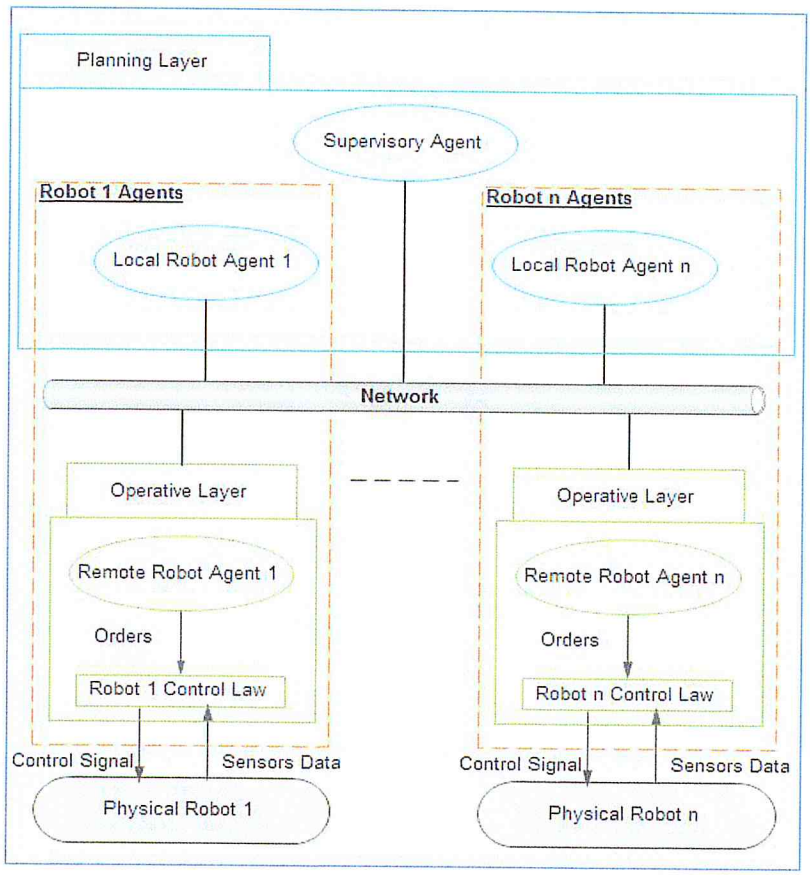


Figure II.8.*Structure de l'architecture de contrôle proposée* [29].

II.3.1. Supervisory Agent

Cet agent permet à l'opérateur de gérer et d'interagir avec l'architecture globale de contrôle, en configurant le système et les agents robotiques. En outre, SA reçoit les tâches (ensemble d'opérations) à effectuer par les robots du système. Ces derniers les décomposent

en opérations réalisables par les entités robotiques, puis il distribue ces opérations aux agents locaux (aux robots).

De plus, cet agent prendra en charge la résolution des problèmes de conflit de décisions lorsqu'une collision est détectée lors de l'exécution d'un mouvement entre robots. SA regroupe les modules suivants (figure II.9):

- ***Décision*** : C'est le module principal de l'agent, il permet au SA de prendre les décisions selon le message reçu de la part de tous les agents locaux ou de la part de l'utilisateur à travers de son interface Home/Machine.
- ***Communication module*** **:** il gère la communication .Il contient un sous-module d'analyse de messages, coder/décoder et analyser les messages reçus.
- ***Activation and Control module*** **:** il effectue une analyse continuelle de message que l'agent SA reçoit, et active les comportements qui leur correspondent. Il gère aussi l'affichage des messages (Fin de tâche succès/échec, robot tombé en panne, …).
- ***Configuration module*** **:** permet à l'operateur de configurer l'agent et d'introduire les informations dans la base des connaissances.
- ***Knowledge Base*** **:** C'est la base des connaissances locales de l'agent, elle rassemble toutes les informations concernant les agents de l'architecture (ID_agent, IP_address, Send_Port, Receive_Port, les opérations que les robots peuvent accomplir), et aussi les requêtes à échanger.

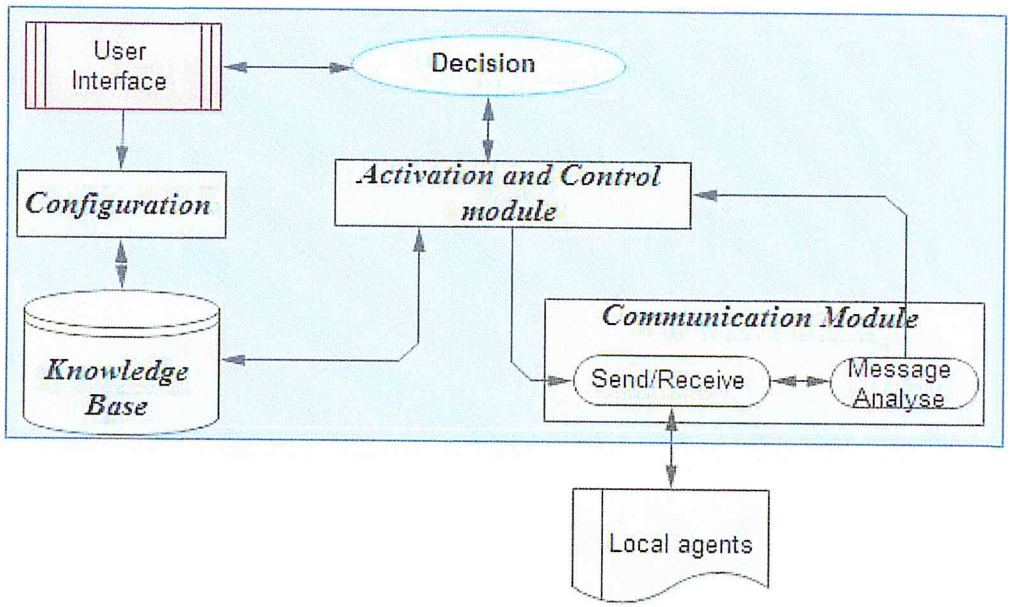


Figure II.9. *Architecture interne du SA.*

II.3.2. Robots agents

Le système multi-robots sera contrôlé à distance. À cet effet, deux agents ont été assignés, pour chaque robot, deux agents différents qui communiquent par des sockets TCP/IP tel que montré par la figure II.10.

a. Local agent

Il rassemble tous les modules nécessaires pour la coopération/coordination avec les autres agents dans la phase de la planification des mouvements (des opérations venantes de l'agent SA). Le résultat obtenu (plan d'opérations) sera envoyé vers RA pour l'exécution. De plus, cet agent obtient et affiche sur son IHM les feedbacks (information des capteurs, rapports d'exécution, etc.) issus de RA.

- ***Trajectoires generation module*** **:** il est responsable de générer la trajectoire (d'une opération) à suivre par le robot afin d'atteindre son objectif, puis l'envoyer vers RA (sous forme d'angles) pour l'exécution. On note que notre travail sera intégré dans ce module.
- ***Local plan generation and Scheduling*** **:** ce module s'occupe de l'ordonnancement des opérations envoyées par SA, les enfile au sein de son plan local dans un ordre bien précis, selon des règles en optimisant le temps total d'exécution.

Architecture de contrôle multi-agents proposée pour le SMR

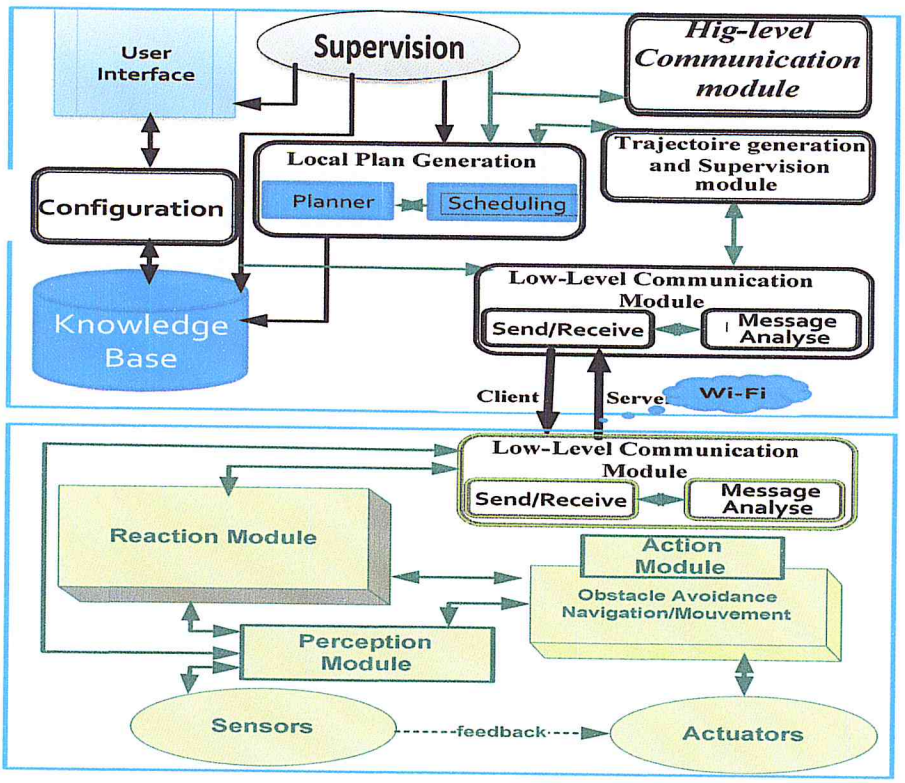


Figure II.10. Architecture interne des *Robots agents*.

b. Remote agent (RA)

- Cet agent est chargé d'exécuter le plan d'opérations envoyé par LA et, d'envoyer, à la fin d'exécution, le rapport d'état d'exécution. Cet agent a comme rôle aussi d'envoyer les informations issues de tous les capteurs équipant le robot. Il est composé des modules suivants :
- ***Réaction module*** **:** c'est le noyau de l'agent qui permet de réagir aux différents événements internes et externes en déclenchant le comportement adéquat.
- ***Perception module*** **:** il permet de percevoir l'environnement où évolue le robot utilisant les capteurs (si le robot est manipulateur mobile, et utilisant les capteurs d'efforts si le robot est un manipulateur).
- ***Action module*** **:** ce module a une relation directe avec le *Perception* module. Il regroupe toutes les compétences de l'agent (mouvement, évitement d'obstacles (manipulateur mobile), arrêt, …) en générant les consignes à envoyer aux actionneurs en fonction de la situation dans laquelle se trouve le robot.

II.4. Conclusion

Depuis leur apparition, les SMA ont pris une place de plus en plus importante en robotique. Ce domaine fournit un ensemble de concepts adaptés à la description et à la compréhension de phénomènes ou comportements distribués pour les SMR. Il permet également de valider et de mettre en œuvre ces concepts au travers de plates-formes informatiques liées au domaine robotique.

Le chapitre suivant est une introduction à l'étude cinématique des corps, et à la modélisation géométrique des RMs, ainsi un aperçu sur les approches existantes pour PTs.

Chapitre III :

Modélisation géométrique et approches de planification de trajectoires

III.1. Introduction

Avant d'introduire la méthode proposée pour la planification de trajectoires dans un SMR, nous présentons dans un premier temps quelques notions de mécanique du mouvement et les notions pour décrire le mouvement des corps. Nous préciserons ensuite ces notions dans le cas des RMs.

Une étude vient par la suite pour résumer les approches existantes dans le cadre de PTs avec un positionnement de notre problème qui clôtura le chapitre.

III.2.Modélisation géométrique des robots

La modélisation géométrique est l'ensemble des outils mathématiques, numériques et informatiques qui combinés permettent de construire un modèle virtuel (ou modèle informatique) d'un objet réel. Cet objet peut être plus ou moins complexe, plus ou moins schématisé. Il peut être le fruit de l'imagination, d'une tendance ou plutôt une solution plus ou moins exacte d'un problème physique donné, voire un compromis entre les deux [33].

III.2.1. Mouvement des corps

L'étude de la cinématique du mouvement se différencie de celle de la dynamique. La cinématique est l'étude des mouvements sans se soucier des causes qui les produisent. Concrètement, la cinématique s'intéresse à l'évolution des corps au cours du temps et plus précisément elle étudie la position, la vitesse, l'accélération et ses dérivées, elle intervient au niveau de la planification de chemin tandis que la dynamique étudie les relations entre les forces et les mouvements qu'elles produisent. Dans nos travaux, nous nous sommes uniquement intéressés à la planification avec des contraintes cinématiques. Avant d'étudier le mouvement d'un corps, nous introduisons les outils géométriques.

III.2.1.1. Situation d'un corps

Considérons un référentiel R_W de système d'axes $(o_w, \vec{o}_x, \vec{o}_y, \vec{o}_z)$,(voir la figure III.1).La situation d'un corps rigide B est la localisation spatiale de ce corps (position et orientation). Pour définir la situation X d'un corps B dans l'espace, nous lui associons un repère R_B.

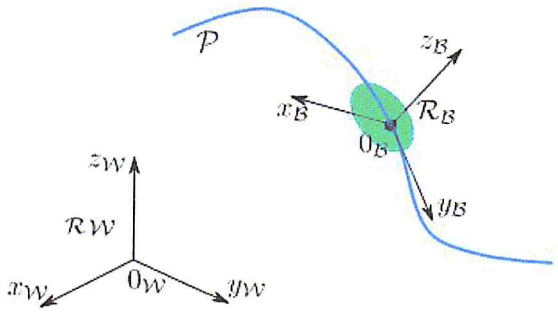


Figure III.1. *Localisation d'un corps dans l'espace 3D.*

La situation d'un corps B peut donc être modélisée par de nombreuses combinaisons des représentations de la position et de l'orientation et de leurs variantes. Six paramètres (3 translations, 3 rotations) définissent cette transformation.

La matrice de transformation homogène T_{WB} est couramment utilisée. Elle permet d'exprimer la transformation entre un repère de référence, par exemple R_W, et un repère R_B associé au corps B :

$$T_{WB} = \begin{bmatrix} R & P \\ 0\ 1 & 0\ 1 \end{bmatrix} \qquad \text{(III.1)}$$

Où R est une matrice de rotation 3×3 appartenant au groupe spécial orthogonal des rotations SO(3) et P une matrice 3×1 représentant la position du point O_B. La matrice de transformation homogène 4×4 associée appartient alors au groupe des déplacements $SE(3)=R^3 \times SO(3)$. Soit bun point appartenant au corps B et positionné par rapport au repère R_B par le vecteur :

$$^{B}P_b = [X_b Y_b Z_b]^T \qquad \text{(III.2)}$$

La position de ce point b dans le repère R_w est :

$$^{w}P_b = T_{WB}\,{}^{B}P_b \qquad \text{(III.3)}$$

Soit R_D un repère appartenant à un corps D et défini par rapport au repère R_B par la transformation T_{BD}. La situation T_{WD} du repère R_D par rapport à R_w est :

$$T_{WD} = T_{WB}\,T_{BD} \qquad \text{(III.4)}$$

II.2.1.2. Chemin

C'est un itinéraire représenté géométriquement dans un plan de déplacement d'une posture initiale vers une posture finale. Il est composé de points (positions ou des postures) intermédiaires connectant les points initial et final. Ils peuvent être connectés par des segments (chemins géométriques) ou paramétrés par des courbes, sous formes de fonctions polynomiales (chemins paramétrés) [34].

II.2.1.3. Trajectoire

C'est une fonction définie dans l'espace des postures, en fonction du temps (t) de sorte à respecter des contraintes liées à la tâche. Elle décrit une succession de déplacements comme le montre la figure III.2, définis comme des configurations de bras manipulateurs, des postures de robots mobiles ou de Manipulateurs mobiles. Les vitesses ainsi que les accélérations y faisant également partie sont déduites par différentiation. Fondamentalement, le problème de génération de trajectoire consiste à trouver une relation entre deux éléments, appartenant aux deux différents domaines: le temps et l'espace. Le but étant de trouver des trajectoires pour des chemins spécifiés à priori, remplissant un/plusieurs critère(s) (comme le temps de parcours (T_{fin}-T_{init}) minimum par exemple) [34] .

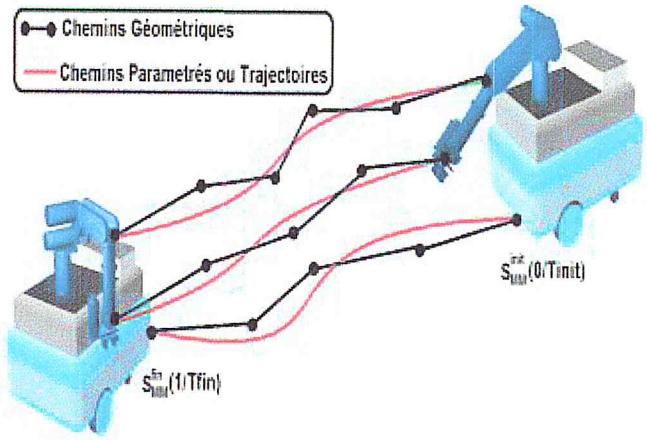


Figure III.2. *Planification de chemins et génération de trajectoire* [34].

III.2.2. Mouvement des robots

La modélisation géométrique et cinématique des robots et la planification de mouvement sont largement décrites dans la littérature [33].

L'environnement dans lequel évolue le robot est l'espace de travail W (figure III.3) avec $W \subset R^3$.Cet espace comprend les objets sur lesquels le robot doit agir, les obstacles et les humains. Un robot est un ensemble de corps rigides reliés par des liaisons cinématiques (e.g. liaison glissière, pivot, rotule...).

Chaque liaison constitue un ou plusieurs degrés de liberté du robot. L'ensemble de ces liaisons définissent la chaine cinématique. Une extrémité de la chaine cinématique est appelée base. Les extrémités du ou des bras manipulateurs sont appelés organe(s) effecteur(s). Lozano-Pérez [42] propose de représenter la posture du robot par un vecteur dont la dimension correspond au nombre de degrés de liberté du robot. L'espace de la représentation paramétrique des postures du robot est nommé espace des configurations. Par exemple, un objet en déplacement libre dans l'espace, possède six degrés de liberté (trois translations et trois rotations), son espace des configurations est donc de dimension six.

Nous noterons R_w, le repère définissant l'origine de l'espace de travail W. Le repère de la base du robot sera noté R_{rob}.

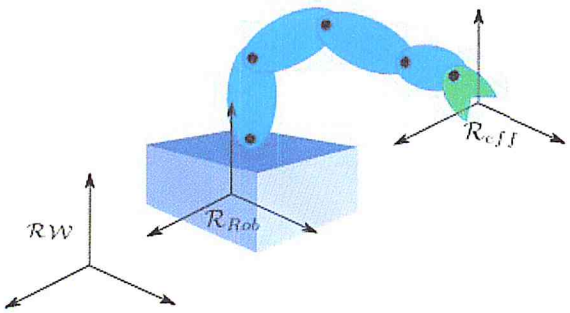


Figure III.3. *Repères associés au robot* [33].

Le modèle géométrique direct d'un robot, défini par la fonction f_{MGD}, permet d'exprimer la situation de OT X (appelée également coordonnées opérationnelles) du robot manipulateur par rapport à R_{rob} en fonction de sa configuration q:

$$X = \mathbf{f_{MGD}}(q) \qquad \text{(III.5)}$$

Le modèle géométrique inverse permet de déterminer une configuration (pas forcément unique) du robot q en fonction de la situation de l'organe terminal :

$$q = \mathbf{f_{MGI}}(X) \qquad \text{(III.6)}$$

III.3. Différentes approches de planification de trajectoires

Dans cette section, nous résumons les approches les plus significatives apparues dans le domaine. Elles sont généralement divisées en trois classes : les méthodes globales, les méthodes locales et les méthodes mixtes [20].

- **Approches locales :**

On parle de méthodes locales car la connaissance d'un obstacle, aussi précise soit elle, ne renseigne en aucun cas quand à la place disponible dans le voisinage de l'obstacle, ou entre celui-ci et tout autre obstacle. Elle lui donne également un comportement local car les décisions de choix et de changement de direction, sont guidées essentiellement par l'obstacle rencontré. Les méthodes locales n'ont besoin que d'une connaissance partielle de l'espace de travail. Elles cherchent une trajectoire au fur et à mesure que le robot avance. La trajectoire initiale est généralement représentée par une ligne droite liant la position initiale à la position finale dans l'espace de configurations. A chaque étape de l'algorithme, le processus vérifie

l'existence d'un éventuel obstacle. Et dans un tel cas, la trajectoire est alors modifiée localement afin d'éviter la collision.

Le principe consiste à engendrer progressivement des incrémentations de déplacements pour permettre de se rapprocher du but en utilisant une information locale sur les contraintes imposées par l'environnement. Cette information peut être issue, par exemple, de capteurs de distances ou de capteurs tactiles ou encore, directement, par un algorithme de calcul de distances.

On reconnaît à ces méthodes les avantages suivants:

- Rapidité pour l'obtention d'une solution,
- Possibilité de prendre en compte rapidement les obstacles imprévus,
- Obtention d'une trajectoire continue (non segmentée) qu'il n'est pas nécessaire de lisser.

Cependant, dans certains cas ces méthodes échouent alors qu'une solution existe.

Exemple : Méthode des champs de potentiels

La plus connue et la plus populaire des méthodes locales est la méthode des champs de potentiels. A l'origine, cette méthode a été développée dans le but de réaliser l'évitement d'obstacles en ligne. Elle est utilisée quand le robot n'a, à priori, aucune représentation des obstacles dans l'espace de travail, mais doit cependant les localiser et les éviter durant l'exécution de son mouvement. Cette technique introduit des champs de forces artificiels dans lesquels se déplace le manipulateur. Les obstacles sont considérés comme des points à haut potentiel et le point qui correspond à la configuration finale (le but) comme un point à bas potentiel.

Le robot est ainsi placé dans ce champ constitué de forces répulsives (sortant des obstacles) et de forces attractives dirigées vers le point but final. Sous l'action combinée des forces répulsives et attractives, le robot se déplacera toujours d'un point à haut potentiel vers un point à bas potentiel. Les champs de potentiel servent à diriger le manipulateur dans son environnement, afin qu'il atteigne son objectif. Le manipulateur est modélisé par des points de contrôle qui sont influencés par ces champs de potentiel. Un point de contrôle attaché à l'effecteur est poussé vers son objectif par un champ de potentiel attractif. Aussi, des points

de contrôle attachés le long du RM sont repoussés des obstacles. La figure III.4 explique mieux le principe.

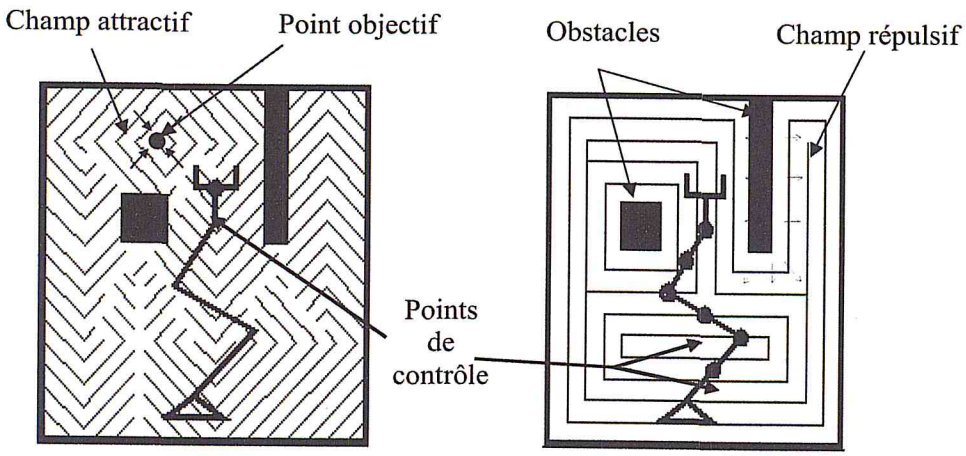


Figure III.4.*Modèle du robot et de l'environnement dans la méthode des champs de potentiel* [20].

Le robot va être considéré comme étant placé dans :

- Un champ de potentiel attractif dont la source est le but.
- Un champ de potentiel répulsif dont la source est l'obstacle.

- **Approches globales :**

Les méthodes globales sont basées sur une connaissance complète de l'environnement dans lequel le robot évolue. L'espace libre est généralement représenté dans l'espace des configurations du robot (appelé aussi C-Space) par une série de paramètres donnant la position et l'orientation de l'effecteur. L'avantage de formuler le problème de planification dans le C-Space est qu'il devient équivalent à la navigation d'un point objet. Afin de pouvoir représenter l'espace des configurations libres, les méthodes globales ont besoin d'une transformation des obstacles à partir de leurs représentations cartésiennes en une représentation dans l'espace de configuration du robot. Il est ainsi possible de déterminer les configurations qui emmènent le robot de la configuration initiale à la configuration finale souhaitée, en évitant celles qui provoquent des collisions.

Exemple : Décomposition en cellules

Cette méthode est peut-être la technique la plus utilisée dans la recherche de trajectoires. Elle est basée sur la division de l'espace de travail en cellules. Les cellules représentent l'espace libre dans lequel le robot peut se mouvoir et ainsi l'espace occupé par les obstacles. Si une cellule est partiellement occupée, elle est alors divisée en cellules de tailles plus réduites, jusqu'à obtenir des cellules ou vides ou occupées, ou enfin, obtenir la précision souhaitée. Une fois cette subdivision réalisée, l'espace peut être décrit par un graphe qui représente la relation d'adjacence entre les cellules. Ensuite, la recherche de trajectoires est basée sur l'exploitation de ce graphe.

La première phase de la méthode consiste en la décomposition en cellules de l'espace libre (figure III.5.b). La seconde phase est la construction du graphe d'adjacence (figure III.5.c) qui représente la relation entre les cellules. La recherche d'une trajectoire consiste à trouver un chemin dans le graphe représentant l'espace libre entre la cellule contenant la configuration initiale et celle contenant la configuration finale.

Dans la figure III.5.d, on peut observer les cellules (zone hachurée) par lesquelles peut passer le robot. A ce stade, différentes trajectoires du robot sont possibles, aussi il faut définir un critère de choix afin d'obtenir une trajectoire optimale au sens de ce critère. Dans le cas de la figure III.5.e, le critère utilisé consiste à faire passer le robot le plus loin possible des obstacles (à mi-distance des obstacles). Les butées et les limites de l'espace de configuration sont considérées comme des obstacles. La trajectoire obtenue est représentée par la ligne brisée de la figure III.5.e.

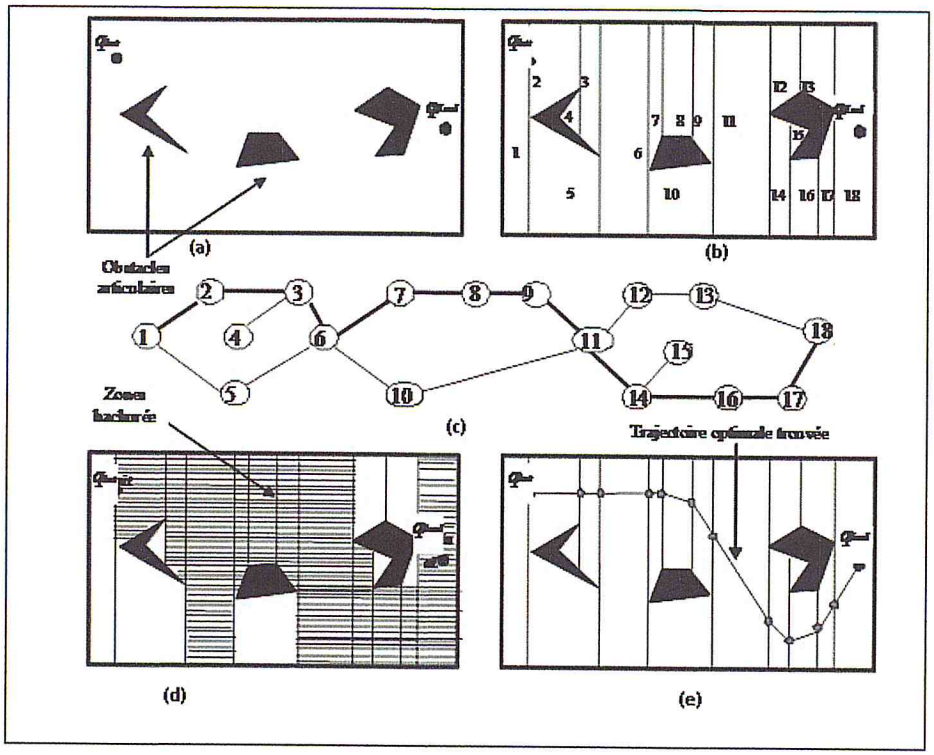


Figure III. 5. *Décomposition en cellules* [20].

- **Approches mixtes :**

La présence des minima locaux reste la cause la plus importante d'inefficacité dans les méthodes locales, d'autres part les approches globales sont très coûteuses en temps de calcul et en conséquence elles ne sont pas adaptées pour des applications en temps réel. C'est pourquoi plusieurs chercheurs ont essayé de combiner les deux techniques mentionnées précédemment.

Les méthodes mixtes opèrent en deux phases : la première phase, réalisée hors ligne et appliquée généralement aux trois premiers d.d.l, est nettement inspirée des méthodes globales, et la deuxième phase est inspirée des méthodes locales et exécutée en ligne et traitant l'intégralité des d.d.l. Dans la première phase, une planification de type globale est réalisée : construction du graphe représentant l'environnement et recherche d'une trajectoire dans ce graphe. La principale différence repose sur le fait que le graphe est très simplifié et, par conséquent, le temps de calcul et l'espace mémoire utilisés sont réduits considérablement. Dans la deuxième phase, une méthode locale est utilisée pour la navigation du robot et la trajectoire de référence est donc modifiée au fur et à mesure que le robot avance. La transition dans le graphe entre les différents nœuds ou cellules est calculée de différentes

façons, par exemple, à l'aide d'une fonction de potentiel. Ceci permet d'améliorer la trajectoire calculée initialement.

III.4.Travaux existants

En robotique, on est souvent amené à devoir planifier des trajectoires pour permettre à un robot de se déplacer d'un point initial à un point final. La PTs est un sujet qui est souvent traité dans le domaine scientifique. Il existe donc plusieurs algorithmes différents et des travaux permettant de réaliser une telle tâche. Dans cette section, nous présentons certaines de ces méthodes en expliquant brièvement leur principe ainsi que leurs avantages et leurs inconvénients.

Généralement les techniques utilisées dans le cadre de planification de trajectoires par les chercheurs sont classées en deux catégories, des méthodes globales et des méthodes locales. Le choix de celles à choisir est vraiment en fonction des contraintes décrites par l'environnement ou la structure physique du robot lui-même, et l'enchainement des idées présentées dans l'algorithme proposé. Venant maintenant aux travaux qui faisant coopérer les deux grandes classes, ils sont cités ci-après [20]:

• Dans [43], une méthode mixte avec un planificateur local utilisant les champs artificiels de potentiel et un planificateur global (Les HEKM : Hierarchical Extended Kohonen Map) coopèrent pour résoudre le problème de planification de trajectoire.

• [43], [48], [46], et [49] ont proposé de générer globalement la trajectoire comme un résultat du processus du prétraitement avec un planificateur local. En particulier, [45], [46] et [44] utilisent une Self Organization Map (SOM), et [46] utilise une variante dynamique de SOM (DSOM) basée sur le développement des Growing Neural Gaz Network [47].

Les solutions mentionnées auparavant évitent, dans la plupart des cas la convergence vers des minima locaux ou permettent de s'échapper de ces minima. Par contre, elles ne sont pas adaptées à travailler en temps réel. En fait, le temps nécessaire pour la phase de reconnaissance hors ligne de l'environnement pénalise fortement leur application en ligne, notamment lorsque l'environnement est fortement encombré. Ces méthodes deviennent, elles aussi, très lentes quand l'espace de travail où le robot évolue change continuellement.

Nous parlons maintenant de quelques études qui ont attiré notre attention dans le cadre de l'IA, basées sur le principe de la LF. Cette méthode qui permet d'obtenir des trajectoires

progressives avec évitement d'obstacles, et qui été largement utilisée pour des robots mobiles, comme pour l'exemple présenté dans l'article de Ferdinand Piette [17], qui consiste à implémenter un système flou pour un robot mobile très basique, possédant deux roues motorisés plus une roue libre ainsi que trois systèmes de capteurs permettant de déterminer la distance du robot aux obstacles situés en face, à droite et à gauche. Le but du système flou, c'est de déterminer la vitesse des roues droite et gauche en fonction de la distance des obstacles autour du robot. Faisant pour cette méthode floue des règles d'inférences englobant les 8 cas possibles voir la figure III.6 pour la déduction des solutions qui permettent au robot de se mouvoir en toute sécurité via les entrées sorties floues schématisées dans la figure III.7 ci-dessous.

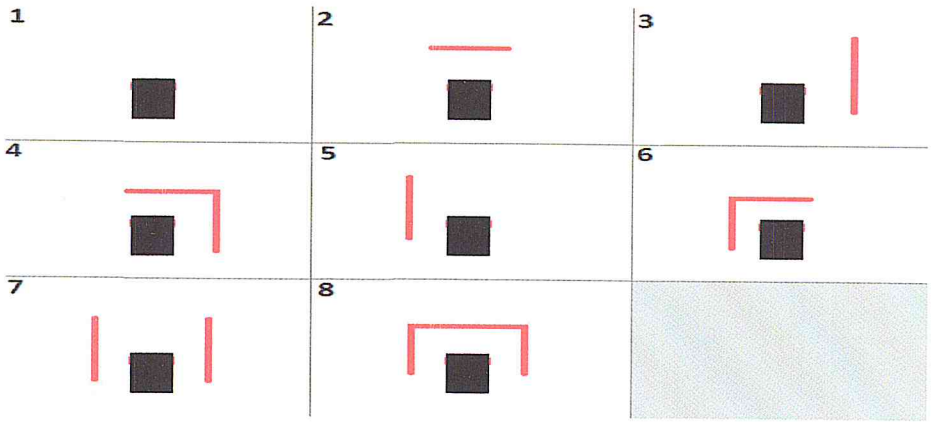


Figure III.6. *Cas possibles pour l'évitement d'obstacles* [17].

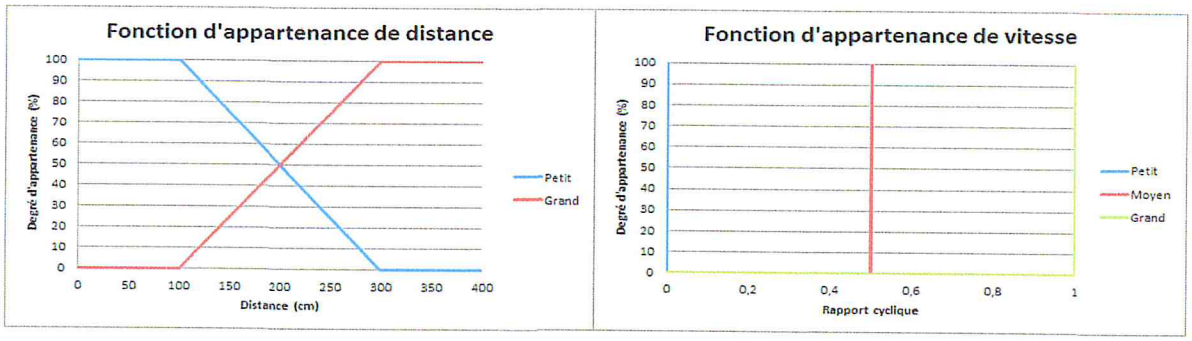


Figure III.7. *Entrées sorties du système flou* [17].

Avec cet exemple, la PTs est simple à implémenter pour un robot mobile en utilisant la LF, mais reste un moyen limité pour d'autres types de robots.

Un autre travail utilisant la LF fait par Moussaoui et son équipe [36], basé sur une méthode de contrôle d'un robot de 6DDL avec évitement d'obstacles, consiste à utiliser un contrôleur flou capable d'évaluer le vecteur d'évitement $V_{evitment}$ correspondant à la position actuelle d'obstacle. La figure III.8 résume le travail.

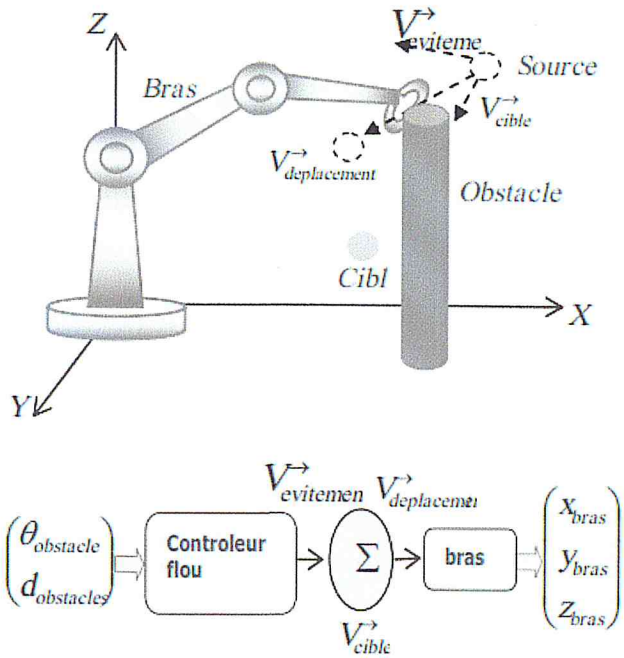


Figure III.8. *Entrées sorties du contrôleur flou* [36].

Cette technique est très utile pour résoudre le problème de collisions, mais une combinaison avec d'autres méthodes pour la génération des postures décrivant la trajectoire du robot est nécessaire, comme le cas du travail réalisé par Kermiche [37]. Où l'objectif du travail était de proposer des algorithmes de génération automatique dans l'espace libre d'un chemin entre la posture initiale et finale par l'emploi d'une méthode non géométrique de planification dite méthode du champ de forces faisant appel à la technique de LF, qui conduit à la génération automatique de trajectoires sans collisions.

Dans cette méthode, le mobile va être considéré comme étant placé dans :

- Un champ de force attractive dont la source est la cible.
- Un champ de force répulsive dont la source est l'obstacle.

A partir de ces forces on détermine l'orientation du bras, comme la figure III.9 l'indique.

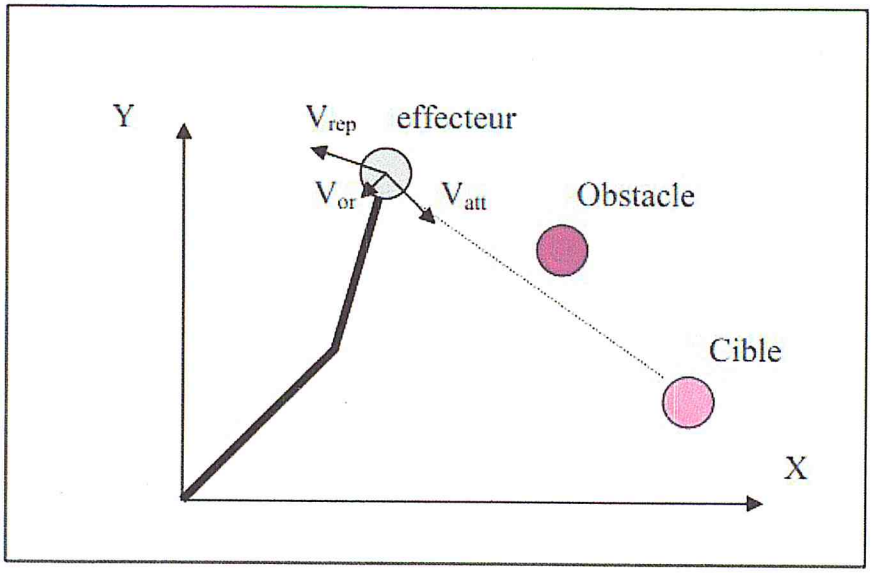


Figure III.9. *Espace de commande du bras manipulateur* [37].

La force attractive est calculée directement à partir de la position de la cible et pour la force répulsive on utilise un raisonnement par logique floue. Le contrôleur flou utilisé écourte le travail suivi dans la figure III.10 ci-dessous :

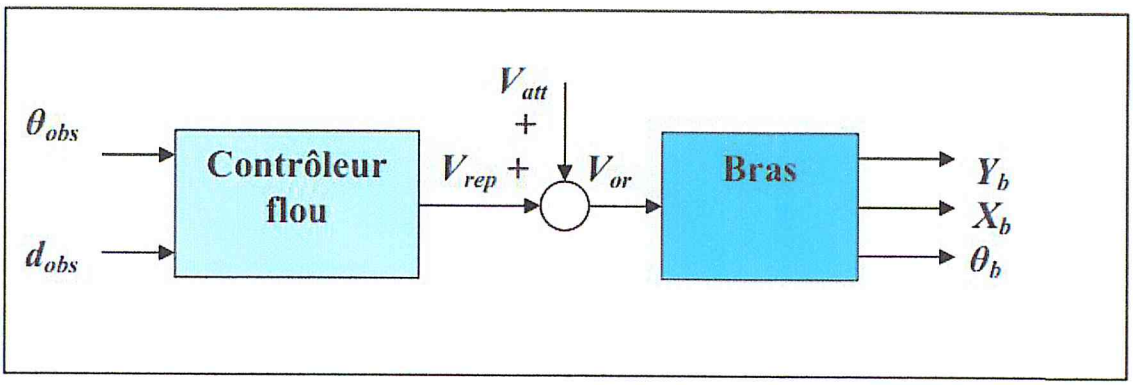


Figure III.10. *Entrées / sorties du contrôleur* [37].

Les approches vues précédemment sont généralement prétraitées de manière à réduire la complexité du problème et le nombre de règles nécessaires. Le formalisme flou, de part ses propriétés d'approximation universel, a la possibilité de décrire la transformation recherchée, mais l'expression sous forme de règles de notre compréhension du mécanisme de navigation ne permet pas de générer une solution acceptable. Pour cela, d'autres travaux sont tournés vers l'apprentissage automatique en contrôle (apprentissage supervisé) [37].

III.5. Positionnement de nos travaux

Le problème de planification de mouvements a suscité de nombreux travaux durant ces dernières années conduisant à un nombre très important d'approches.

Le chapitre suivant s'articule autour du problème de mouvements pour un SMR du point de vue de la PTs sans collisions. Pour cela, nous proposons une méthode basée sur le principe des approches globales et locales ,qui consiste précisément à mettre en œuvre une stratégie de suivi de trajectoires pour un SMM au sein d'une architecture multi-agents de contrôle des SMR de type manipulateur présentée dans le deuxième chapitre. Le but de l'expérience présentée précédemment par les travaux existant est de concevoir un GFTs simple avec un nombre réduit de règles, permettant au robot de se mouvoir avec une PTs hors ligne et avec un test de collisions en ligne, afin d'éviter les collisions avec d'autres RMs travaillant dans le même espace de travail sous la gestion de l'architecture de contrôle multi-agents. Nous allons détailler dans la suite chacune des étapes vues pour atteindre la cible du sujet.

III.6.Conclusion

Ce chapitre présente des méthodes pour la planification de mouvement. Ces dernières années des travaux ont été effectués sur la PTs pour permettre de résoudre des tâches complexes dans de grandes dimensions et dans des temps raisonnables.

Le chapitre suivant va expliquer notre démarche, en détails, en ce qui concerne la PTs sans collisions au sein d'un SMM.

Chapitre IV :

Méthode proposée pour la planification des trajectoires

IV.1 Introduction

L'intérêt de ce chapitre est de proposer une méthode basée sur la LF pour générer la trajectoire d'un RM à 6DDL. Cette méthode est orientée de manière à régler le problème de difficulté et parfois d'absence d'un modèle mathématique pour la modélisation des RMs ayant un nombre de degrés de liberté assez important. Elle permet au bras manipulant de se déplacer pour atteindre un but bien défini.

Par la suite, nous proposons une approche de détection des éventuelles collisions dans un SMR exécutant les trajectoires générées et partageant le même espace de travail.

IV.2.Rappels sur la logique floue

De nos jours, la LF est un axe de recherche important sur lequel se focalisent de nombreux scientifiques. Les bases théoriques de la LF ont été formulées en 1965 par le professeur Lotfi A. Zadeh, de l'Université de Berkeley en Californie. Il a introduit la notion de sous ensemble flou pour fournir un moyen de représentation et de manipulation des connaissances imparfaitement décrites, vagues ou imprécises. A cette époque, la théorie de la logique floue n'a pas été prise au sérieux excepté par quelques experts. Peu de temps après, les Japonais ont commencé à utiliser largement la LF dans des produits industriels et de consommation pour résoudre des problèmes de réglage et de commande [38].

IV.2. 1.Définition (sous-ensemble flou)

Un sous-ensemble flou A dans un univers du discours X est caractérisé par sa fonction d'appartenance $\mu_A(x)$ qui associe à chaque élément x de X une valeur dans l'intervalle des nombres réels [0, 1].

$$\mu_A : x \rightarrow [0,1] \qquad \text{(IV. 1)}$$

IV.2. 2.Opérations sur les sous-ensembles flous

Supposons que A et B sont deux sous-ensembles flous définis dans un univers du discours X par les fonctions d'appartenance μA et μB. On peut définir des opérations ensemblistes telles que l'égalité, l'inclusion, l'intersection, l'union et le complément grâce à des opérations sur les fonctions d'appartenance.

- L'égalité :

$$\forall x \in X \quad \mu_A(x) = \mu_B(x) \tag{IV.2}$$

- L'inclusion :

$$\forall x \in X \quad \mu_A(x) \leq \mu_B(x) \tag{IV.3}$$

- L'intersection :

$$\forall x \in X \quad \mu_{A\cap B} = \min\left(\mu_A(x), \mu_B(x)\right) = \mu_A(x) \wedge \mu_B(x) \tag{IV.4}$$

- L'union :

$$\forall x \in X \mu_{A\cup B} = \max\left(\mu_A(x), \mu_B(x)\right) = \mu_A(x) \vee \mu_B(x) \tag{IV.5}$$

- Le complément :

$$\forall x \in X \quad \mu A^c = 1 - \mu_A(x) \tag{IV.6}$$

IV.2.3.Définition (variable linguistique)

Dans une partition floue forte (voir figure IV.1), chaque ensemble correspond à un concept linguistique, par exemple *Très faible*, *Faible*, *Moyen*, *Elevé*, *Très élevé*. Pour le raisonnement les variables sont manipulées par les termes linguistiques ainsi définis, les ensembles flous assurent la correspondance avec l'univers numérique [16].

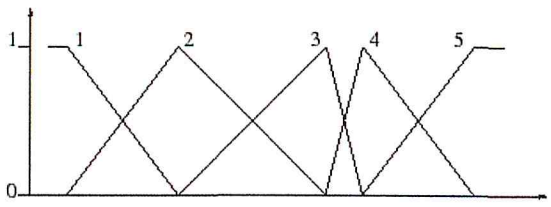


Figure IV.1. *Exemple de partition floue forte* [16].

IV.2.4. Règle floue

Une règle floue est de la forme Si je rencontre telle situation Alors j'en tire telle conclusion. La situation, appelée prémisse ou antécédent de la règle, est définie par une

combinaison de relations de la forme x est A pour chacune des composantes du vecteur d'entrée. La partie conclusion de la règle est appelée conséquence, ou encore simplement conclusion [16].

IV.2.5. Système d'inférence flou

Système d'inférence flou (SIF) : est formé de trois blocs comme indiqué sur la figure IV.2. Le premier, l'étage de fuzzification transforme les valeurs numériques en degrés d'appartenance aux différents ensembles flous de la partition. Le second bloc est le moteur d'inférence, constitué de l'ensemble des règles. Enfin, un étage de défuzzification permet, si nécessaire, d'inférer une valeur nette, utilisable en commande par exemple, à partir du résultat de l'agrégation des règles [16].

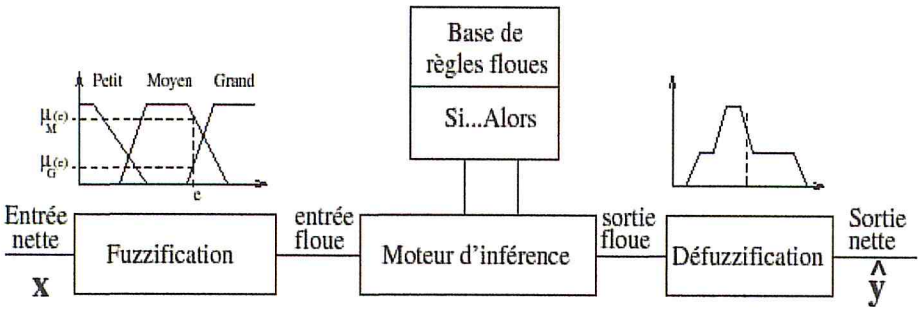


Figure IV.2. *Système d'inférence flou* [16].

IV.3 Présentation de l'approche proposée

Généralement l'utilisation des outils mathématiques de modélisation est appropriée et justifiée pour les systèmes bien définis. Mais, quand la complexité augmente, ces outils deviennent moins efficaces. Le traitement des systèmes complexes nécessite souvent la manipulation d'informations incertaines. Un robot doit posséder des capacités de perception et de mouvement nécessaires pour qu'il puisse exécuter ses tâches dans son environnement.

Les techniques de l'IA basées sur la LF sont considérées comme une solution très intéressante pour les systèmes complexes où il est difficile d'établir un modèle mathématique, La LF fournit un meilleur moyen d'automatiser les expertises humaines. Elle permet aux experts humains d'exprimer leurs connaissances qui sont généralement difficiles à mettre en application lors de la conception de systèmes de traitement classiques, de façon naturelle et

que le minimum des règles est suffisant pour exprimer les concepts. Donc gain de temps et d'espace mémoire, ce qui donne une rapidité considérable aux moteurs d'inférence.

La méthode de génération de trajectoires proposée pour un bras manipulateur consiste à construire un GFTs robuste, simple et adéquat pour la planification des différents déplacements exécutés par un RM au sein d'un SMR. On présente dans ce qui suit son architecture, les informations acquises et fournies par ce dernier.

IV.3.1. Description géométrique du bras

Le modèle d'un robot est une représentation répétable, interprétable et simulable d'un robot, c'est-à-dire un ensemble d'expressions mathématiques qui permettent de décrire le robot. Et parmi les types de modèles, on a le model géométrique vu dans le chapitre précédent, qui est une relation mathématique entre les coordonnées articulaires et opérationnelles, ce qui est bien connu par deux espaces dans la littérature :

- Espace articulaire est l'espace qui a pour référence le repère lié à chaque articulation motorisée du robot. Les coordonnées associées sont appelées coordonnées articulaires.
- Espace opérationnel est l'espace qui a pour référence le repère lié à l'organe terminal du robot. Les coordonnées associées sont appelées coordonnées opérationnelles.

La structure de notre bras est la Structure Ouverte Simple (St.O.S) dont la quelle chaque corps a au plus deux articulations. La St.O.S permet d'amener l'OT dans une situation (position et orientation) donnée. Elle est la plus utilisée et a pour avantage essentiel d'augmenter la rigidité ainsi que la précision. Le bras manipulateur à étudier est un robot articulé à 6 DDL. 6DDL sont suffisants pour positionner l'OT dans une position et orientation complètes dans l'espace. La cinématique du porteur est de type RRR et le poignet comporte trois rotations comme la figure IV.3 l'indique.

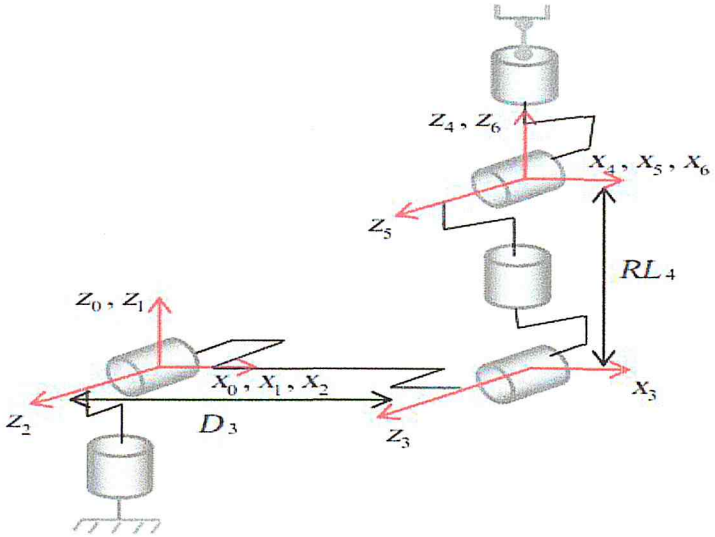


Figure IV.3. *Placement des* repères *et notation d'un RM* [36].

IV.3.2. Générateur de trajectoire flou proposé

L'objectif est la création d'un planificateur flou inspiré des systèmes d'inférences existants, capable d'évaluer les coordonnées articulaires q_i, i=1,…, 6, d'un bras manipulateur qui permettent au manipulateur de se déplacer d'une position initiale $P_{int}(x, y, z)$ à une position finale $P_{fin}(x, y, z)$ tout en évitant l'utilisation du MGI et en respectant les contraintes (limites) d'articulations et des vitesses articulaires.

Notre planificateur flou utilise seulement le MGD du robot ce qui nous a permis d'éviter l'utilisation du MGI (difficile à calculer, pas d'exactitude dans les résultats, etc.).

En réalité, il n'y a pas de solution générale pour ce modèle. De plus s'il y a une solution, elle n'est pas unique. Cependant, l'imposition de contraintes supplémentaires permet de dégager la solution la plus adaptée. Donc notre objectif est de contourner ce problème par l'utilisation d'une approche basée sur le résonnement humain.

La trajectoire générée avec le GFTs dans l'espace articulaire est soumise aux diverses contraintes. Les représentations mathématiques des restrictions considérées dans ce travail sont présentées dans les deux équations suivantes :

- Contrainte d'articulation :

$$\begin{cases} t \in [0, T] \\ q_{i\_min} \leq q_i(t) \leq q_{i\_max} \end{cases} \quad \text{(IV.7)}$$

Où q_{i_min}, q_{i_max}les limites de l'articulation *i*. T représente le temps global de mouvement de l'articulation q_i.

- Contraintes de vitesse :

$$\begin{cases} t \in [0,T] \\ V_i(t) \leq V_{i\_max} \\ V_i(0) = 0,\ V_i(T) = 0 \end{cases} \quad \text{(IV. 8)}$$

Où V_{i_max} est la vitesse angulaire maximale de l'articulation *i*.

La figure IV.4 montre un bras manipulateur dans une position initiale (P_{int}) qui vient de déplacer un objet vers une position cible (P_{cible}). Donc comme est indiqué dans le schéma, le robot doit minimiser l'erreur d'angle et l'erreur d'altitude pour atteindre la cible. Par conséquent, les articulations ayant une influence sur l'erreur d'angle vont mouvoir le sens gauche afin de minimiser cette erreur. Les articulations ayant une influence sur l'erreur de l'altitude vont mouvoir le sens down. Notre GFTs est basé sur le même principe, il a des connaissances aprioris sur le type de mouvement de chaque articulation (gauche, droit, monte, descend), donc il génère les valeurs de ces articulations dans le sens où il minimise les erreurs de l'altitude, et d'angle, ainsi que la distance entre la pince et la cible (il s'arrête si la distance $\leq \varepsilon$, avecεcompris entre (1-5 cm)).

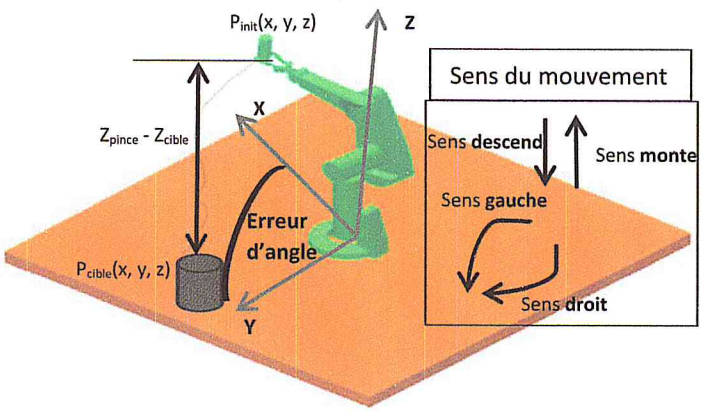


Figure IV.4.*Position du bras par rapport à la cible.*

Le générateur de trajectoire flou proposé est illustré dans la figure IV.5. Il possède trois entrées et deux sorties :

- les entrées sont la position de la cible, la position de la pince, et l'ensemble des coordonnées articulaires initiales du robot manipulateur.
- Les sorties sont l'angle de déplacement et sa vitesse.

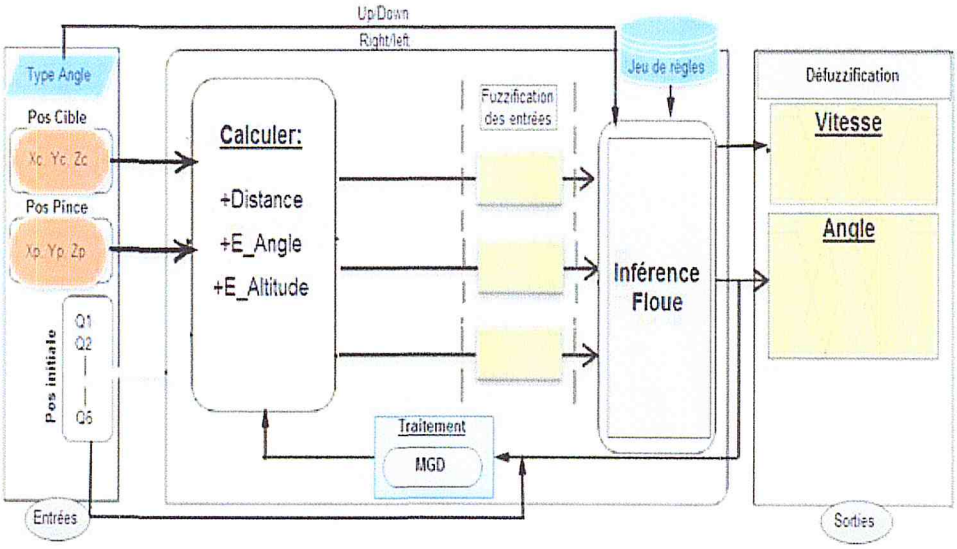


Figure IV.5.*Générateur flou proposé pour la génération de trajectoires.*

Le planificateur flou ou bien dit générateur de trajectoires possède comme connaissance locale définie préalablement le type de mouvement de chaque articulation, il a aussi comme entrées *(i)* la position de la cible, la position de la pince et les q_i (i=0, .., 6) initiales du bras.

Les données d'entrées passent par un module de calcul, ce module permet de calculer les erreurs d'angle et d'altitude avec la distance séparant la pince de la cible. Ces grandeurs sont nécessaires pour la phase de fuzzification, ces valeurs sont calculées comme suit :

- **Distance :** c'est la longueur qui sépare la position de la pince à la position de la cible dans un espace de trois dimensions, décrit par la formule suivante :

$$\textbf{Distance} = \sqrt{(x_p - x_c)^2 + (y_p - y_c)^2 + (z_p - z_c)^2} \quad (IV.9)$$

- **E-Altitude :** présente aussi une grandeur qui exprime l'écart entre la composante z de la position de la pince et celle de la position de la cible.

$$\textbf{E- Altitude} = z_p - z_c (IV.10)$$

- **E-Angle :** est la différence entre l'angle formé par l'axe x et l'axe y repérant la posture prise par la pince et celle de la cible, exprimé par la formule ci-dessous :

$$\textbf{E-Angle} = \tan^{-1}\left(\frac{Yp}{Xp}\right) * \left(180/\pi\right) - \tan^{-1}\left(\frac{Yc}{Xc}\right) * \left(180/\pi\right) \quad (IV.11)$$

Où (X_p, Y_p, Z_p) et (X_c, Y_c, Z_c) sont les cordonnées de la pince et la cible selon l'axe x, l'axe y et l'axe z respectivement.

Le processus général de génération de trajectoire utilisant la LF est présenté dans l'organigramme de la figure IV.6.

Le planificateur flou fournit comme sortie (après la phase de défuzzification) un angle et une vitesse angulaire nécessaire pour le déplacement d'un segment parmi les six segments constituant le bras manipulateur tout en évaluant leurs valeurs à partir des règles d'inférences emmagasinées dans la base du contrôleur et selon l'écart de la distance , d'altitude, et d'angle calculés à chaque traitement effectué pour estimer la différence qui sépare la pince de sa cible.

Selon la nature du mouvement de l'articulation, les angles fournis en sortie sont de deux types :

- type droit ou gauche : pour minimiser l'écart d'angle.
- type monté ou descend : pour minimiser l'écart d'altitude.

Par la suite, ces nouvelles coordonnées articulaires seront injectées au module MGD qui sert à calculer la nouvelle position de la pince. Ce module met à jour les coordonnées articulaires, par l'ajout de cet angle aux coordonnées articulaires initiales des articulations de même type afin de calculer les coordonnées cartésiennes $New\_P_{pinc}(x, y, z)$ de la pince qui seront ensuite fournies comme une nouvelle entrée au module calcul. Le processus poursuit son exécution jusqu'à ce que la position de la pince soit très proche de la position de la cible (Distance très petite [1-5] cm).Cette distance de précaution est pour ajuster l'orientation de la pince afin de saisir ou déposer l'objet (la cible).

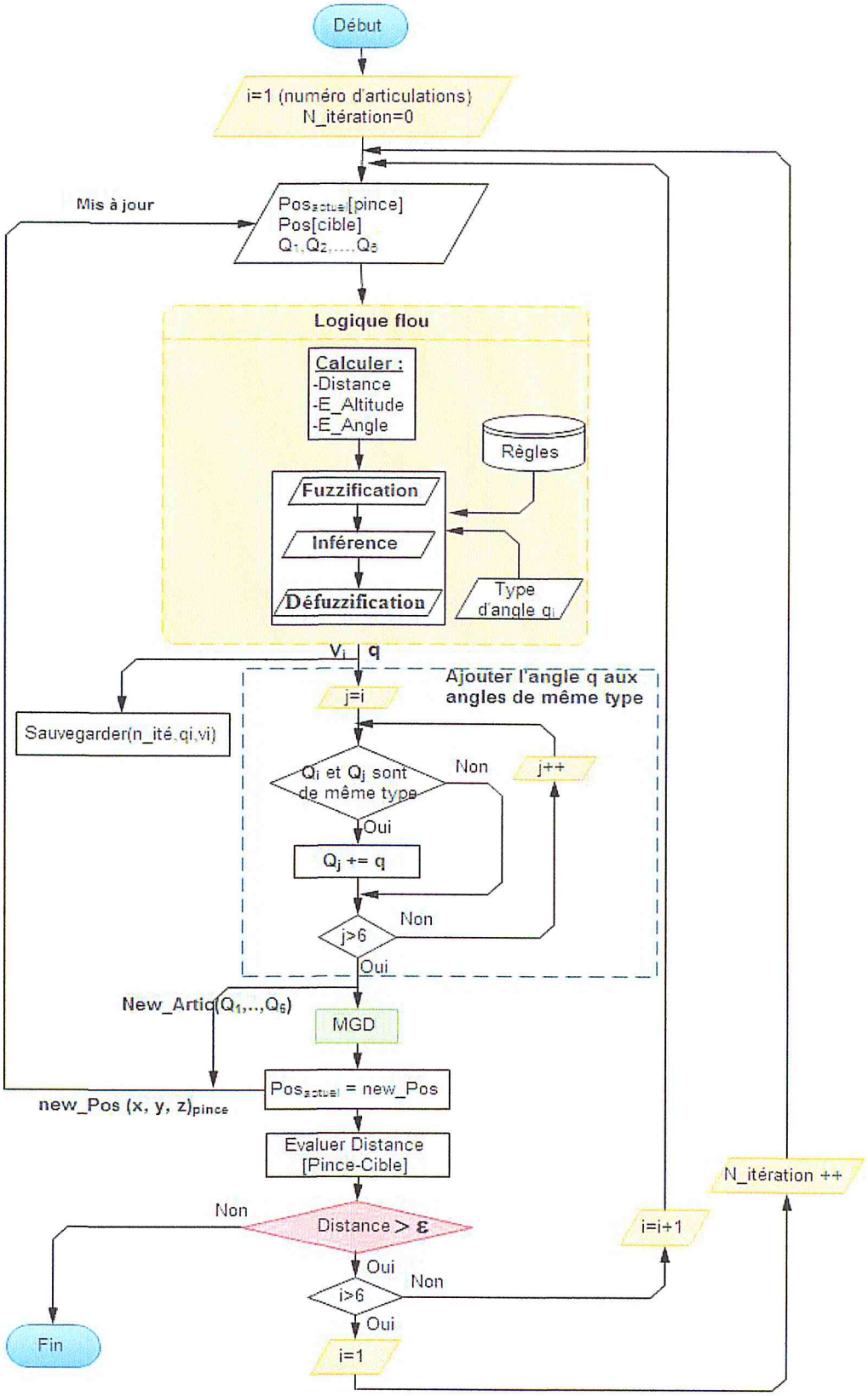


Figure IV.6.*Processus général de la génération de trajectoires.*

Par la suite, on va détailler les trois phases principales de la LF avec des schémas descriptifs des sous-ensembles flous des entrées/sorties du planificateur flou.

IV.3.3. Fuzzification

Cette première étape consiste à déterminer le degré d'appartenance de chaque variable d'entrée à chaque état. Celui-ci est déterminé à l'aide des fonctions d'appartenances.

Dans cette phase on a utilisé deux types de fonctions d'appartenances :*(i)* Trapèze, et *(ii)* Triangle. Le choix de la fonction d'appartenance dépend de la nature des données et du problème. Les sous-ensembles flous des variables d'entrées sont présentés comme suit:

- **Ecart de distance**

Comme la figure IV.7 l'indique, la variable linguistique possède trois termes linguistiques.

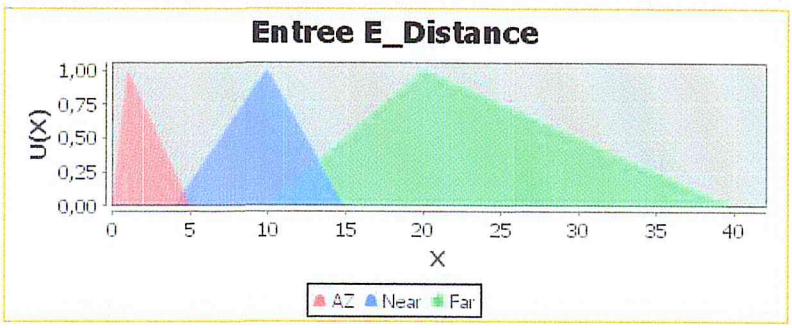


Figure IV.7. *Sous-ensembles flous de l'écart de distance.*

AZ : approximation à zéro.

Near: proche.

Far: loin.

- **L'écart d'angle**

Cette variable décrit les sous-ensembles flous suivants dans un domaine de discours compris entre [-180,180], voir la figue IV.8.

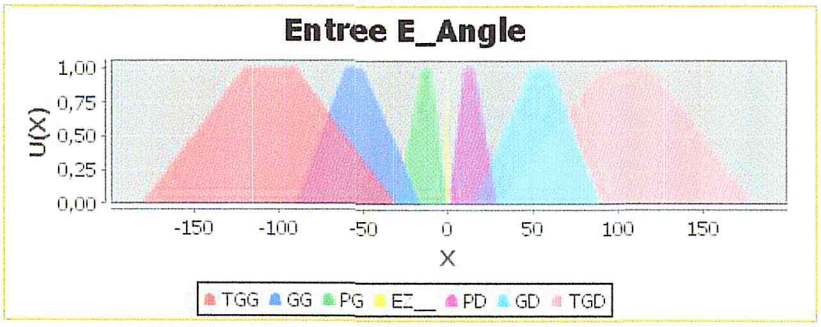


Figure IV.8. *Sous-ensembles flous de l'écart d'angle.*

TGG : très grand à gauche.

GG :grand à gauche.

PG :petit à gauche.

EZ__ :environnement proche du zéro.

PD : petit à droite.

GD : grand à droite.

TGD : très grand à droite.

- **L'écart de l'altitude**

La figure IV.9 ci-dessous, contient sept termes linguistiques présentés dans un intervalle variant de [-90,90].

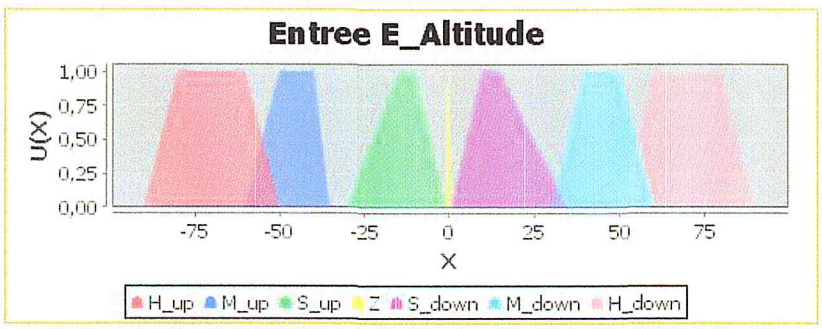


Figure IV.9. *Sous-ensembles flous de l'écart d'altitude.*

H_Down :grand en bas.

M_Down: moyen en bas.

S_Down: petit en bas.

Z : environnement proche du zéro.

H_UP: petit en haut.

M_UP: moyen en haut.

S_UP: grand en haut.

IV.3.4 Inférences

Après avoir fuzzifier les variables d'entrées, et l'établissement des règles liant les entrées aux sorties, les degrés d'appartenance obtenus de chaque variable à chaque état permettent d'appliquer les règles floues qui ont été préalablement définies. Le degré d'appartenance des variables de sortie à chaque état est ainsi obtenu. En d'autres termes l'inférence lie les grandeurs mesurées transformées en variables linguistiques à l'aide de la Fuzzification, à la variable de sortie exprimée comme variable linguistique.

On peut distinguer deux genres de règles d'inférences:

- Inférence avec une seule règle.
- Inférence avec plusieurs règles.

Basées sur le choix d'opérateur (ET), (OU) et l'implication floue (ALORS), il existe plusieurs méthodes d'inférence [38] telles que (*Mamdani*, *Sugeno*, …, etc).Nous utilisons dans ce travail la méthode de (Mamdani).

Un contrôleur flou prend généralement la forme d'une série de règles «Si-alors». L'inférence est basée sur des opérations min et max afin d'effectuer l'inférence des règles et l'opérateur max pour l'agrégation des règles.

Le jeu de règles pour notre GFTs, est décrit sous forme d'un tableau (voir le tableau V.1 au-dessous), dont chaque ligne présente une règle d'inférence, et si plusieurs entrées sont activées, la règle prend la forme (**Si** val_{e1} **AND** val_{e2}…val_{eN} **Alors**val_{s1},val_{s2},….Val_{sM}) avec M et N par ordre présentant le nombre de variables linguistiques activées d'entrées et de sorties.

	Entrées			*Sorties*		
Règle N°	**E_Angle**	**E_Altitude**	**E_Distance**	**Vitesse**	**Q_1 (L/R)**	**Q_2 (UP/Down)**
1	TGG	-	-	G_Vitesse	GR	-
2	GG	-	-	M_Vitesse	MR	-
3	PG	-	-	P_Vitesse	PR	-
4	EZ__	-	-	Z_Vitesse	EZ	-
5	PD	-	-	P_Vitesse	PL	-
6	GD	-	-	M_Vitesse	ML	-
7	TGD	-	-	G_Vitesse	GL	-
8	-	H_UP	-		-	G_UP
9	-	M_UP	-		-	M_UP
10	-	S_UP	-		-	P_UP
11	-	S_Down	-		-	P_Down
12	-	M_Down	-		-	M_Down
13	-	H_Down	-		-	G_Down
14	-	-	AZ	Z_Vitesse	-	E_Z
15	-	S_UP	Near	P_Vitesse	-	P_UP
16	-	P_Down	Near	P_Vitesse		P_Down
17	TGD	-	Far	G_Vitesse	GL	-
18	TGG	-	Far	G_Vitesse	GR	-

Table IV.1.*Règles d'inférences du GFTs.*

IV.3.5. Défuzzification

La sortie du planificateur est en général une grandeur continue, prenant sa valeur dans un intervalle bien défini afin de respecter les contraintes et les limites décrites dans les deux équations (IV.7 et IV.8).

La défuzzification est le traitement qui permet de définir une correspondance entre le résultat de l'inférence et la grandeur continue fournie en sortie. Cette étape s'effectue toujours à l'aide des fonctions d'appartenance. A partir des degrés d'appartenance, on obtient autant de valeurs qu'il y a d'états. Il existe plusieurs méthodes telles que : (la défuzzification

par centre de maximum, par valeur maximum, et par centre de gravité). Dans notre cas on utilise le centre de gravité.

Le planificateur donne une sortie à la fois (soit : monté/descend ou gauche/droit), tout dépond de l'articulation avec une certaine vitesse. Les sous-ensembles flous des deux types d'angles de sortie et la vitesse sont illustrés dans les figures IV. 10, IV.11 et IV.12.

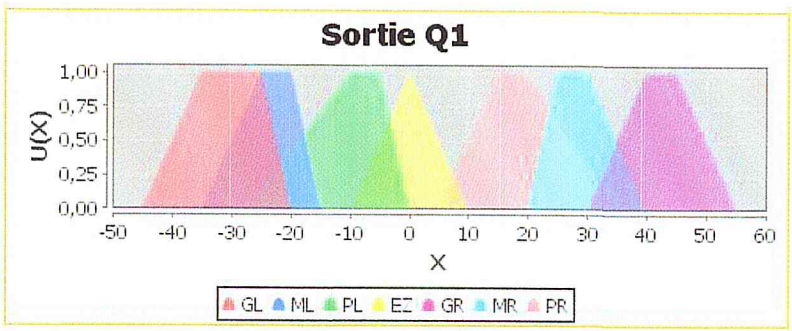


Figure IV.10.*Sous-ensembles flous d'angle de type gauche/droit.*

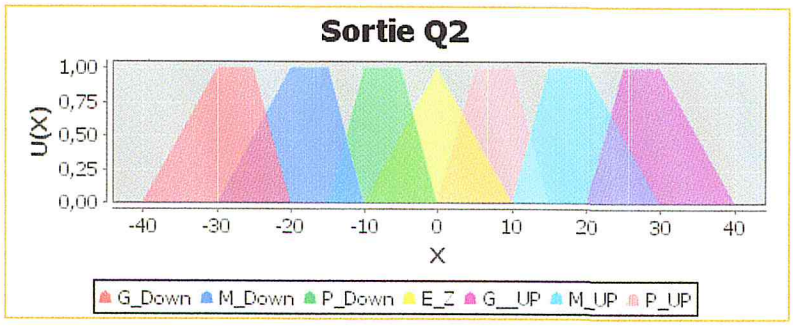


Figure IV.11. *Sous-ensembles flous d'angle de type monté/descend.*

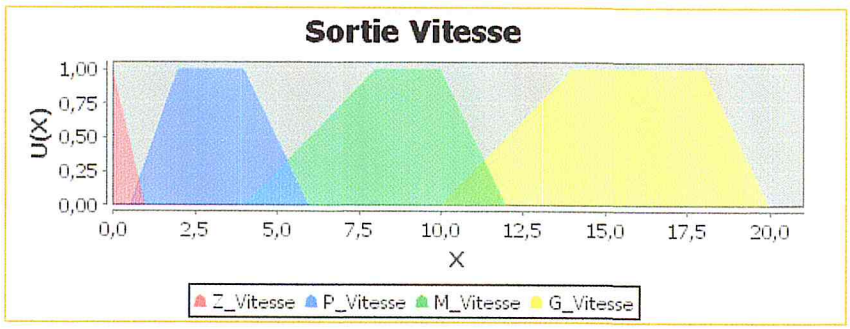


Figure IV.12. *Sous-ensembles flous de la Vitesse.*

IV.4 Evitement de collisions dans un SMR

L'une des missions de base d'un robot est la capacité d'évitement d'obstacles, et l'application de ses mouvements au sein d'un espace sans entrelacement avec d'autres robots ayant des objectifs commun à atteindre. C'est une tâche importante que doivent assurer tous les robots dans un SMR partageant le même espace de travail.

Dans notre travail on s'intéresse aux SMR, donc après la phase de génération de trajectoire, les robots doivent exécuter leurs trajectoires tout en évitant toutes éventuelles collisions entre eux. Dans cette situation, l'échelle de complexité de PTs sans collisions devient très importante avec des robots de type manipulateur 6 DDL, puisque un robot de bras manipulant, est vu comme étant une chaine de corps liés entre eux, ce qui rend le contrôle de son mouvement très difficile à gérer.

Nous considérons seulement les collisions entre les robots (les collisions avec les obstacles statiques n'ont pas été prises en considération dans notre travail). La première phase est la phase de détection de collision, ensuite l'évitement. Nous considérons aussi que les robots bougent une seule articulation à la fois durant l'exécution de leur trajectoire générée par le planificateur flou.

IV.4.1. Principe de détection des collisions utilisant les systèmes multi-agents

Le test de collision commence une fois la planification des trajectoires finie. Après la génération de trajectoires, chaque robot dispose de ses propres angles par lesquels il va atteindre sa cible. Afin d'assurer la détection de collision entre les robots, nous considérons les articulations de chaque robot comme étant des points connectés entre eux par des segments droits. Comme montre la figure IV.13, les points P_0, P_1, P_2, et P_3 sont les points assiégés aux articulations du bras (exemple d'un manipulateur 3ddl). Les segments $Seg_1(P_0, P_1)$, $Seg_2(P_0, P_1)$ et $Seg_3(P_0, P_1)$ sont les segments du bras, chaqu'un est présenté par deux points.

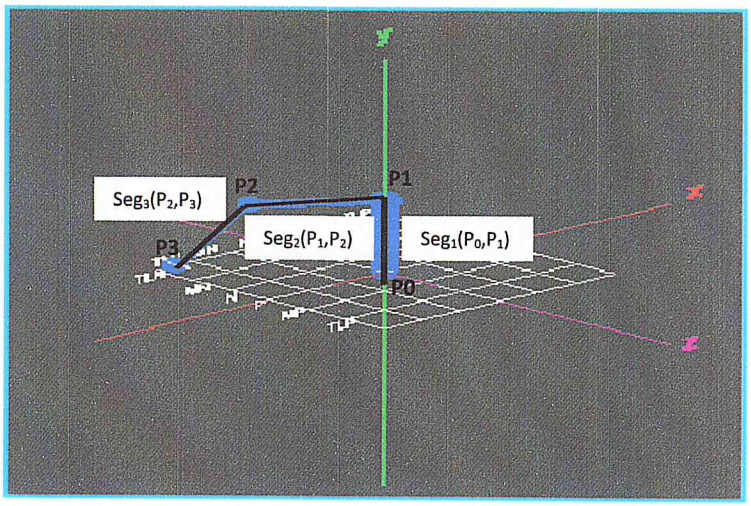


Figure IV.13. *Représentation d'un bras manipulateur par segments droits.*

Donc, la détection de collisions consiste à tester si une collision existe entre deux segments droits dans un espace de trois dimensions, de deux robots manipulateurs. Un segment est représenté mathématiquement par une droite(Δ)exprimée par le système paramétrique ci-dessous :

$$(\Delta)\begin{cases} x = at + \mathrm{x_a} \\ x = bt + \mathrm{y_a}\ t \in [0-1] \\ x = ct + \mathrm{z_a} \end{cases} \qquad \text{(IV.11)}$$

Avec A (x_a ,y_a , z_a) est un point de la droite (Δ) et $\vec{u}\begin{pmatrix} a \\ b \\ c \end{pmatrix}$un vecteur directeur de(Δ) .

Les extrémités du segment droit(Δ)sont deux points d'articulations parmi les six points d'articulations obtenus par la conversion des coordonnées articulaires à l'aide du modèle MGD. Les coordonnés(x, y, z) de chaque articulation à chaque instant *t*(calculé selon le pas choisi) pour la construction des segments droits sont échangées entre les robots via les communications contrôlées par l'architecture du SMA. Le schème de la figure IV.14 illustre un exemple de collision:

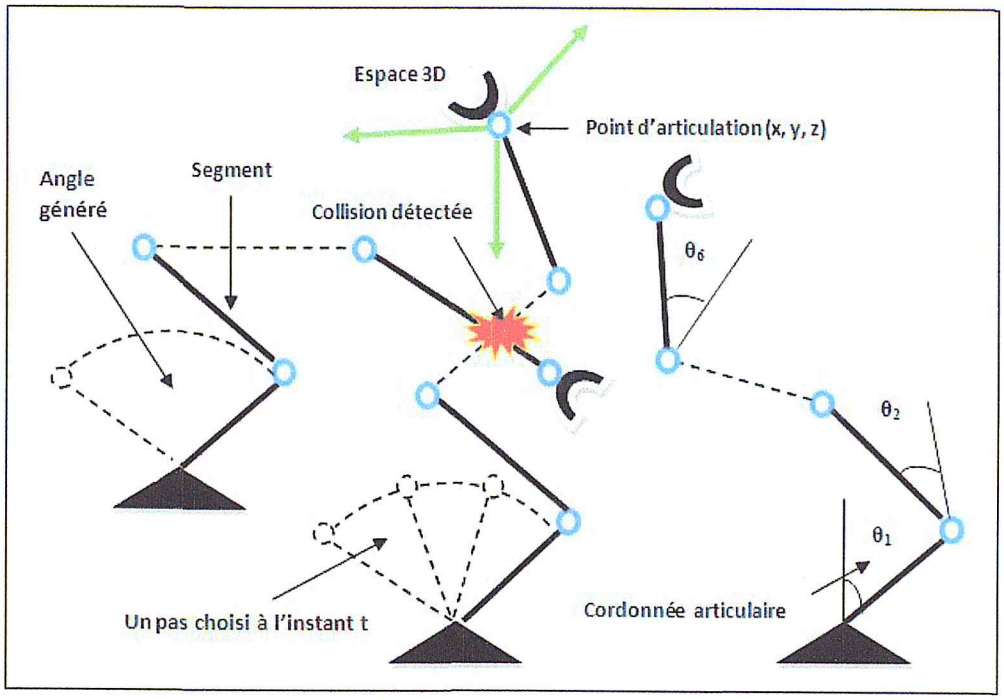


Figure IV.14. *Exemple de détection de collisions entre trois manipulateurs.*

L'un des avantages des SMA est la possibilité d'interagir et de communiquer (coopérer) entre eux. Ces communications vont permettre à chacun d'anticiper quelles seront les actions futures des uns et des autres. Par conséquent, le processus suivi pour la détection de collisions est basé sur ces compétences.

Afin d'assurer la détection de collision, un processus d'interaction entre les agents locaux (agents robotiques) de notre architecture est exécuté. Cette interaction consiste à mettre en relation dynamique des agents locaux, par le biais d'un ensemble d'actions : algorithmes locaux et échange de messages. Cette mise en relation dynamique entre les agents du système permet de résoudre tous les problèmes de conflit de décisions entre agents (par exemple si une collision existe, la résolution de conflit consiste à donner la priorité à un agent parmi eux de passer en premier afin d'éviter la collision).

Le fonctionnement des agents locaux durant cette phase est décrit par un organigramme présenté dans la figure IV.15. Il débute par la connexion avec les autres agents locaux, puis l'introduction du ***pas*** de test et les données générées par le planificateur flou (ce sont les angles des postures prises par le bras manipulant d'une trajectoire), dont chaque itération i présente une posture ou bien une configuration articulaire $Q_i = \{q_1,.., q_j\}$ avec j= {1..6}.Ensuite, l'agent commence ses tests pour chaque décomposition d'un angle Q_{ij} en de

petits pas, afin d'assurer l'absence d'une collision lors de ses mouvements, puisque le déplacement du segment liée à Q_{ij} par ce petit pas, peut l'engendrer. Suite à cette décomposition, une mise à jour est faite aux coordonnées articulaires par l'ajout du pas aux coordonnées de l'état précédent, tout en respectant les contraintes de dépendances entre les angles de même types. Ces nouvelles coordonnées obtenues ou actuelles, sont utilisées pour la détermination des équations de droites de chaque corps constituant le bras manipulant du robot contrôlé par LA.

Si le test indique l'existence d'une collision entre ces dernières et les équations formées par les différentes configurations opérationnelles actuelles récupérées par les agents locaux voisins, une demande pour l'attribution des priorités d'exécutions (voir ces critères dans la section suivante) est établie, et durant cette demande les agents locaux restent bloqués en attendant une décision par SA.

Dès qu'un agent à l'antériorité d'ordonner son RA pour effectuer ce pas, les autres se réactivent par SA à tour de rôle selon la priorité attribuée, sinon si le test de collision est positive, l'agent local poursuit son exécution et ordonne son RA en toute sécurité.

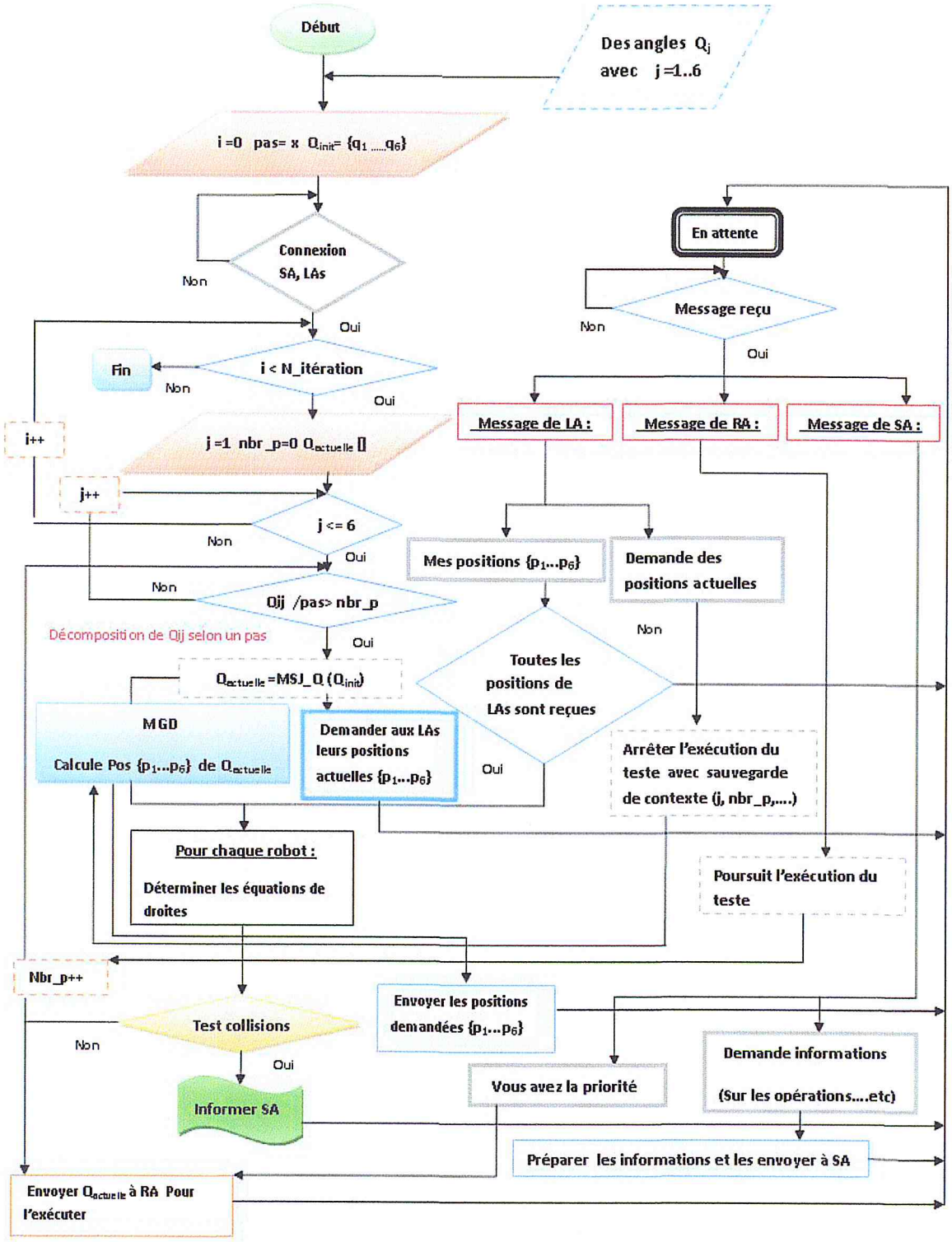


Figure IV.15. *Fonctionnement des agents locaux durant la détection de collision.*

IV.4.2. Principe d'évitement de collisions

La collision entre deux ou plusieurs robots partageant le même espace de travail est vue comme un conflit de décision entre deux agents (les agents des deux robots). Dans notre

système, c'est SA qui gère ce genre de conflits (collisions). Notre principe d'évitement de collision entre deux robots à un certain moment consiste à donner la priorité à un robot pour passer en premier et l'autre robot reste en attente jusqu'à ce que le premier passe, puis il continue son mouvement (synchroniser les mouvements des robots). SA décide qui passe en premier (quel agent prend la priorité ?) utilisant les règles de priorités suivants :

- **Règle 1** : La priorité la plus élevée est accordée au robot exécutant l'opération ayant la plus grande valeur de priorité.

En réalité, La trajectoire exécutée par le robot est la trajectoire d'une opération donnée, et il se trouve que les opérations sont reliées entre eux par des contraintes de précédences. Par conséquent, SA assigne à chaque opération une valeur de priorité qui vaut la somme de la durée de l'opération et les durées de ses opérations successeurs [35].

- **Règle 2 :** La priorité la plus élevée est accordée au robot exécutant l'opération ayant la plus grande durée.
- **Règle 3 :** La priorité la plus élevée est accordée au robot ayant le plus grand nombre d'opérations à exécuter dans son plan local.

L'organigramme de la figure IV.16 détaille le processus de résolution de conflits (exécuté par SA). Le fonctionnement de SA débute par la mise en attente des messages provenant d'un LA. Dès la réception des données relatives à son sous-problème concernant la détection d'une collision qu'il doit gérer, il récupère du message toutes les informations concernant le problème :

- **Id** de l'agent (qui détecte la collision), le nombre d'opérations dans son plan local,…etc
- **Id** de deuxième agent (LA_2).
- Les données de l'opération [Id, la valeur de priorité (val_prior) assignée à cette opération].

Puis, il envoie un message au deuxième agent pour qu'il lui donne ses informations (le nombre d'opérations dans son plan local, et les données de son opération à ce moment. Une fois avoir récupéré toutes les données, il applique la première règle, si le conflit n'est pas résolu il applique la règle 2, sinon il applique la règle 3.Enfin, SA informe l'agent qui va prendre la priorité.

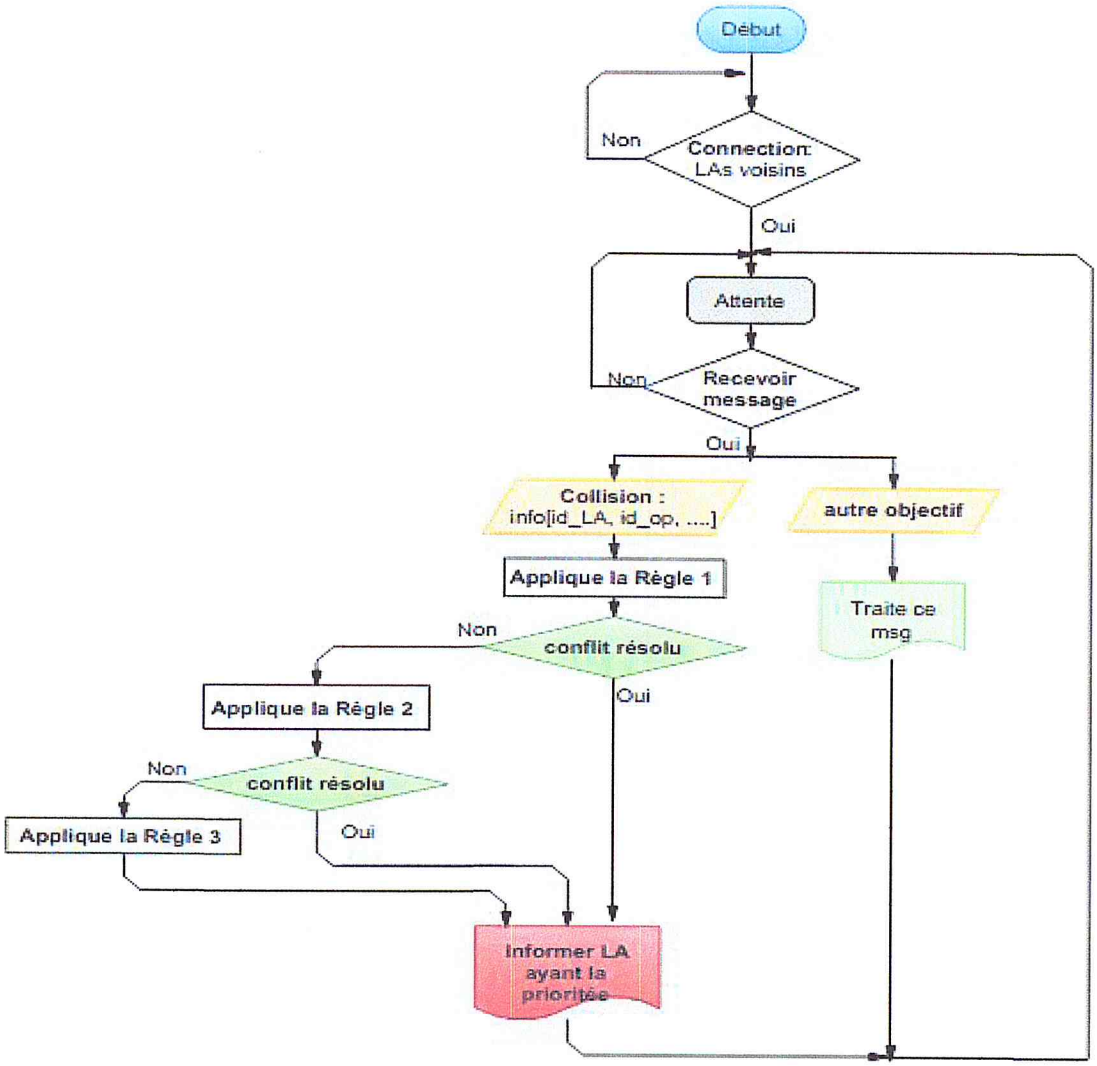


Figure IV.16.*Fonctionnement de l'agent superviseur pour l'évitement des collisions.*

IV.5.Conclusion

Via ce chapitre on a pu faire un tour d'horizon sur le sujet de notre mémoire, par la présentation des différentes procédures entamées dans le cadre de PTs au sein d'un SMR, dont chaque RM est capable d'exécuter la trajectoire sans entrelacement avec d'autres robots, qui présentent les voisins de son environnement de travail, à l'aide de la technique proposée pour l'évitement de collision.

Dans le chapitre suivant, nous présentons l'intégration des méthodes proposées pour l'architecture globale du SMR.

Chapitre V :

Conception et Implémentation

V.1. Introduction

Ce chapitre est consacré à la présentation des différentes étapes suivies pour la conception et l'implémentation de l'architecture multi-agents de contrôle (utilisant deux RMs). À cet effet, le langage de modélisation UML a été utilisé, afin d'illustrer l'architecture globale de notre système.

Le chapitre présente aussi l'intégration de planificateur de trajectoires flou qu'on a proposé dans l'architecture multi-agents de contrôle, décrite dans le chapitre précédent. De plus un processus d'évitements de collisions entre les robots du système sera implémenté à ce stade.

Deux types de diagrammes UML ont été utilisés :

- Diagramme de classes : Il est composé de trois agents décrivant le plan d'opérations nécessaires afin de réaliser les opérations désirées.
- Diagramme de séquence : Ce diagramme présente la communication et les différents comportements échangés entre les différents agents.

Enfin, les principaux résultats obtenus ainsi que leurs interprétations sont présentés à la fin de ce chapitre.

V.2. Architecture de contrôle

La figure V.1 montre l'organisation structurelle et comportementale générale de l'architecture multi-agents proposée dans [26] pour le contrôle de notre SMR constitué de deux robots : un manipulateur mobile (RobuTER/ULM) et un manipulateur (AGVM6).

L'architecture intégrée sur la plateforme Java Agent Development Framework décrite par la suite, consiste en cinq agents (réactifs et délibératifs) (i) SA, (ii) LA1 du robot RobuTER/ULM, (iii) LA2 du robot AGVM6, (iv) RA1 pour RobuTER et (v) RA2 pour AGVM6, où les trois premiers agents présentent les modules du site local ou bien de contrôle. Les deux derniers forment les modules du site distant.

Chaque agent est implémenté en tant qu'un ensemble d'entités concurrentes et communicantes (un ensemble de threads) qui s'exécutent en parallèle de façon autonome.

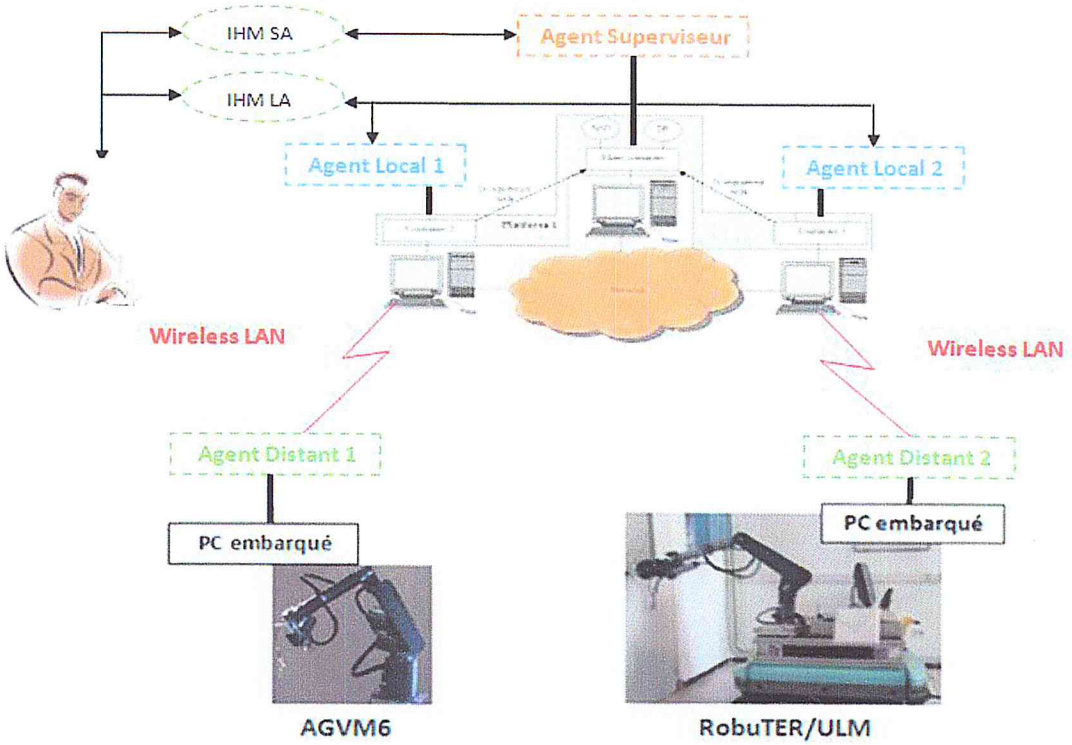


Figure V.1. *Application de l'architecture DMACA sur notre système multi-robots.*

V.2.1. Diagramme de classe

Le diagramme de la figure V.1 présente les différents agents de l'architecture de contrôle. Chacun de ces agents représente une classe du diagramme. Chaque classe comporte un ensemble de compétences pour réaliser l'opération confiée au robot qu'il contrôle.

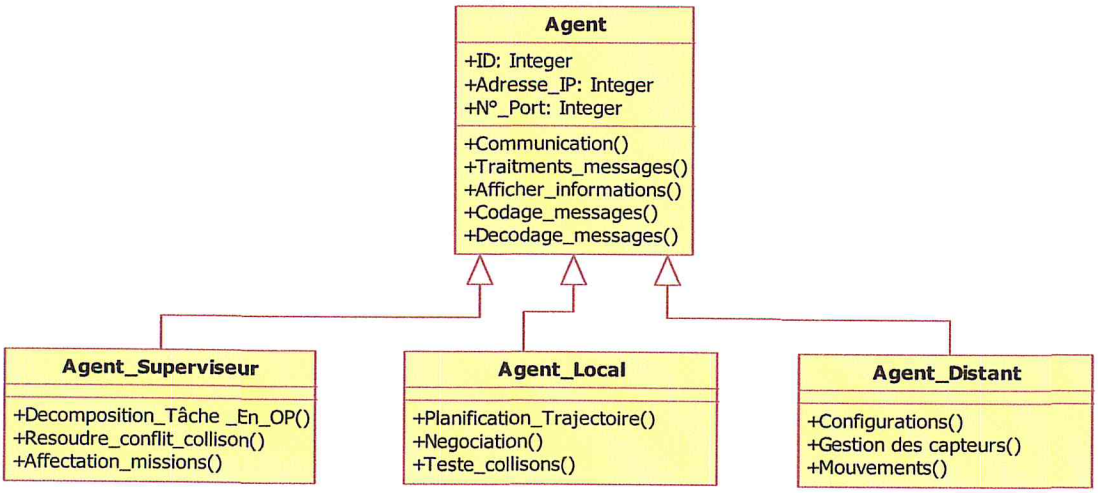


Figure V.2. *Diagramme de classe présentant les différents agents constituant l'architecture DMACA.*

Chaque agent est composé d'un ensemble de modules (compétences et services) fournis par la plateforme JADE. De plus, d'autres fonctionnalités sont intégrées pour résoudre la problématique de ce sujet. Ces différentes fonctionnalités qu'on a apportées sont nécessaires pour la planification de trajectoires sans collisions. Elles utilisent la logique floue et sont données comme suit:

- **Agent superviseur**

  Afin d'interagir avec le système global, cet agent possède des méthodes, ces méthodes appartiennent aux catégories suivantes :

- **Méthodes d'affichages :** C'est pour visualiser des données sur l'interface IHM de l'agent SA
- **Méthode de traitement de messages :** Cette méthode représente l'une des entités de base pour la gestion des problèmes d'accès concurrent lorsque plusieurs agents veulent communiquer avec cet agent. Grace à son comportement intégré (Behaviorcyclic) qui représente des situations d'activations et de blocages pour l'agent afin de traiter un par un les messages reçus, au sein d'une architecture de contrôle. Cette méthode sert à traiter le message selon un objectif bien déterminé dans son entête.

```
Void Message_Handler (msg) {
Obj ←get_header (Msg);
Selon (Obj) {
Cas "Presence":
Informe_Agents (Create_List_agent (Msg.getSender ().getLocalName ()));
```

Cas "Absence":
RemoveAgent_From_List (Msg.getSender ().getLocalName ());
Cas "Collision":
Evaluate_Priority ();
……..
……..
…
}

- **Méthode d'envoi :** cette méthode est généralement exploitée par l'agent lorsque il veut communiquer avec d'autres agents, afin de les informer sur un traitement à effectuer ou bien une donnée demandée à être envoyer nécessaire pour la suite de l'exécution d'un ou d'autre agents.

Void Send_Message (Data, Obj, Receiver) {
Infos←Codage_information (data, Obj);
Msg.setContent (Infos);
Msg.addReciever (Receiver)
Send (Msg);
}

- **Méthodes auxiliaires :** ces méthodes implémentées dans le module agent, servent aux différents traitements effectués par ce dernier. Comme par exemple: (l'évaluation des priorités attribuées aux agents locaux; lorsqu'une collision aura lieu afin de résoudre le problème de conflits concernant l'exécution des mouvements; pour créer la liste des agents existants sur la plateforme, afin d'informer les agents locaux sur leurs voisins; décomposer les tâches en opérations dans un plan selon plusieurs politiques et de les distribuer par la suite sur les agents locaux afin de les traiter ; ainsi que des méthodes pour le codage et le décodage des messages).

Parmi ces méthodes, on trouve : Priority [] Get_Priority (), Void Create_Liste_Agent ()

- **Agents locaux :**

Ces agents ne diffèrent pas de SA, par rapport aux services fournis pour le système comme type. Donc un LA possède lui-même une méthode pour le traitement des messages échangés avec les agents de la plateforme décrite par le pseudo algorithme ci-dessous, sauf que celle-ci prend en considération non seulement la communication avec SA mais aussi avec RA ou même avec d'autres agents locaux afin de contribuer à la PTs.

Void Message_Handler (msg) {
Obj ←get_header (Msg);
Selon (Obj) {
Cas" Get_Pos_To_My_Instant":

```
Inst←get_Content (Msg);
my_pos_for_their_instants (Inst);
Cas "My_Coord_Artic":
Coord []←get_Content (Msg);
If (Teste_Collision (Coord)= true)
Informe_SA ();
Else
Order_RA_to_Run ();
Cas" New_List_Agent":
Create_Contacts (Msg.getSender ().getLocalName ());
Cas "Authority":
Order_RA_to_Run ();
……..
……..
…
}
```

Pour rappel, il y a tout un module constitué de plusieurs méthodes servant au calcul de trajectoires et décrivant le comportement du générateur flou présenté dans le quatrième chapitre (Calculate_trajectory (Pos_{cible}, Pos_{pince}, Q_i []), Calculate_MGD (Q_i [])…..etc), en plus des méthodes dédiées pour les communications avec les agents (Send (Data, Obj, Receiver),…etc).

- **Agents distants**

Du fait de son appartenance au site distant, l'agent distant possède des méthodes pour le décodage des différents messages provenant de son LA situé dans la partie contrôle, et des méthodes pour la communication basées sur le principe des applications client/serveur. De plus, des méthodes pour l'exécution des différentes configurations transmises par LA à des instants bien déterminés, ou pour effectuer une opération en temps réel à l'aide des modèles géométriques existants, pour que le robot meut vers la cible.

V.2.2. Diagrammes de séquence

Ce digramme permet d'exprimer le comportement des agents de notre système tout en montrant l'interaction entre eux.

V.2.2.1. Diagramme de séquence lancement et reconnaissance voisins

Ce diagramme illustre le lancement des différents agents du système par l'opérateur, où chaque LA signale sa présence à SA, pour que ce dernier informe ses voisins sur les agents

actifs au sein de l'environnement. Cette information va aider les agents à bien contribuer au contrôle du système, surtout lors de l'exécution d'une trajectoire ou la réalisation d'une opération d'assemblage, puisque la collaboration et la coordination des tâches représentent l'une des bases du fonctionnement. Ensuite, dès que l'opérateur confirme le lancement de chacun via leurs IHMs, il introduit à SA, le local plan qui concerne les objectifs du travail global commandé à établir par le système.

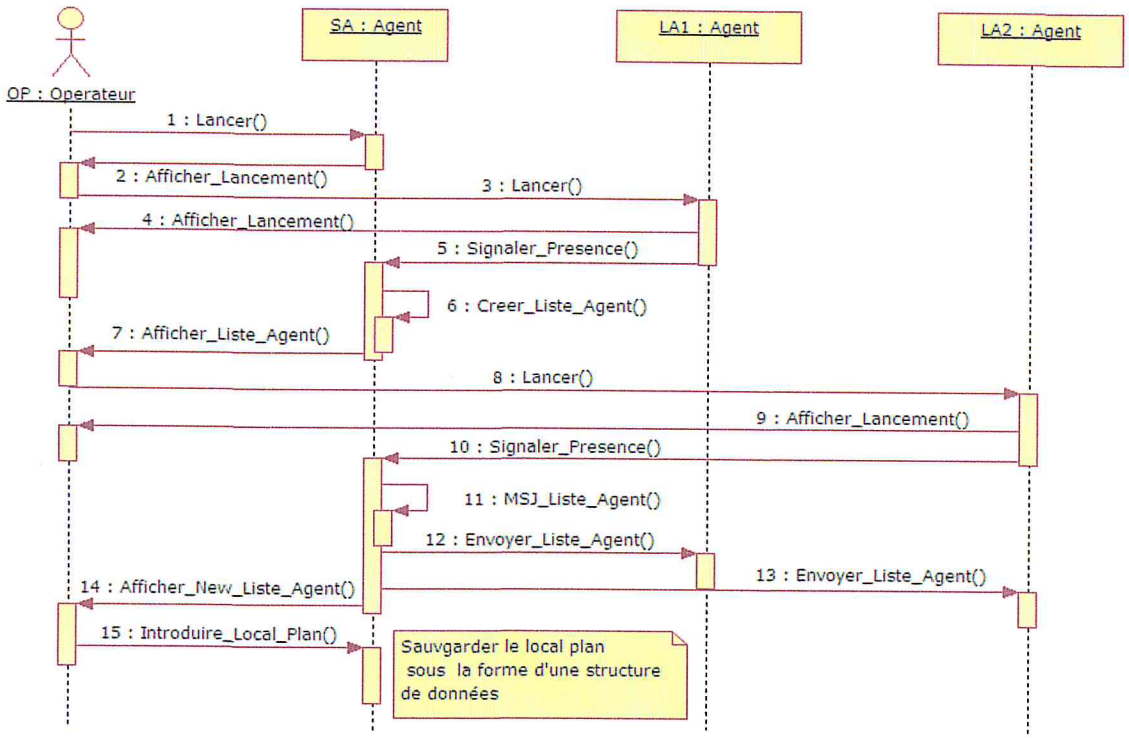


Figure V.3. *Diagramme de séquence lancement et reconnaissance voisins.*

V.2.2.2. Diagrammes de séquences de génération de trajectoires et évitement de collisions

Cette section illustre des diagrammes expliquant le processus d'interaction entre les agents de l'architecture de contrôle, afin de planifier les trajectoires de deux RMs partageant le même espace.

Quand l'opérateur introduit le local plan à SA, des opérations appropriées vont être réparties selon des politiques bien déterminées. Ces dernières sont envoyées aux agents LA1 et LA2 une fois que SA les valide et les réorganise. Par la suite, chaqu'un des agents locaux,

va calculer la trajectoire nécessaire pour la réalisation d'une opération. Le diagramme donné par la figure V.4 explique le déroulement.

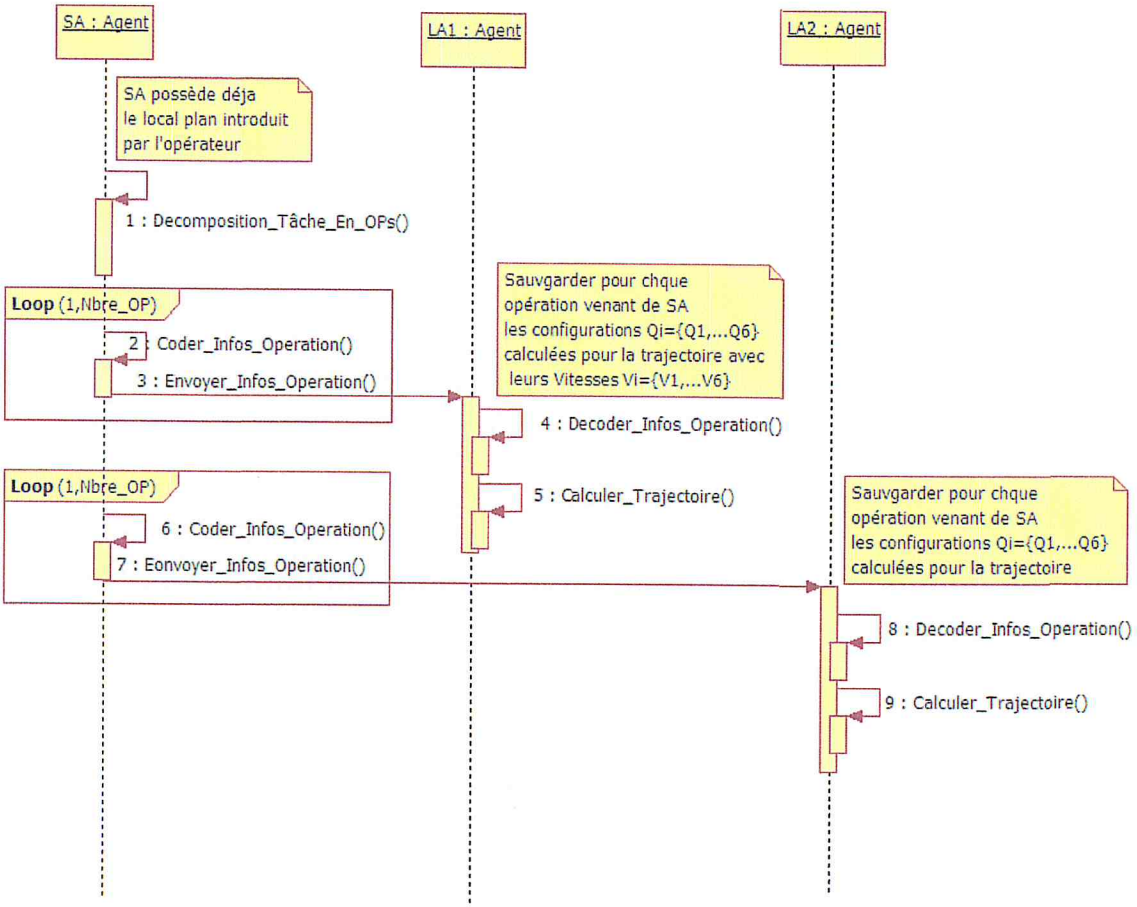


Figure V.4. *Digramme de séquence illustrant le calcul des trajectoires.*

Le diagramme présenté par la figure V.5 expose le processus de résolution des conflits de décisions entre les robots, à partir de leurs agents locaux associés lors de l'existence d'une collision. Le principe est basé sur les communications à temps bien fixé pour l'échange des différentes positions du bras manipulant entre agents locaux afin de tester l'existence d'une collision qui va être résolu par l'attribution d'une antériorité à quelqu'un par SA.

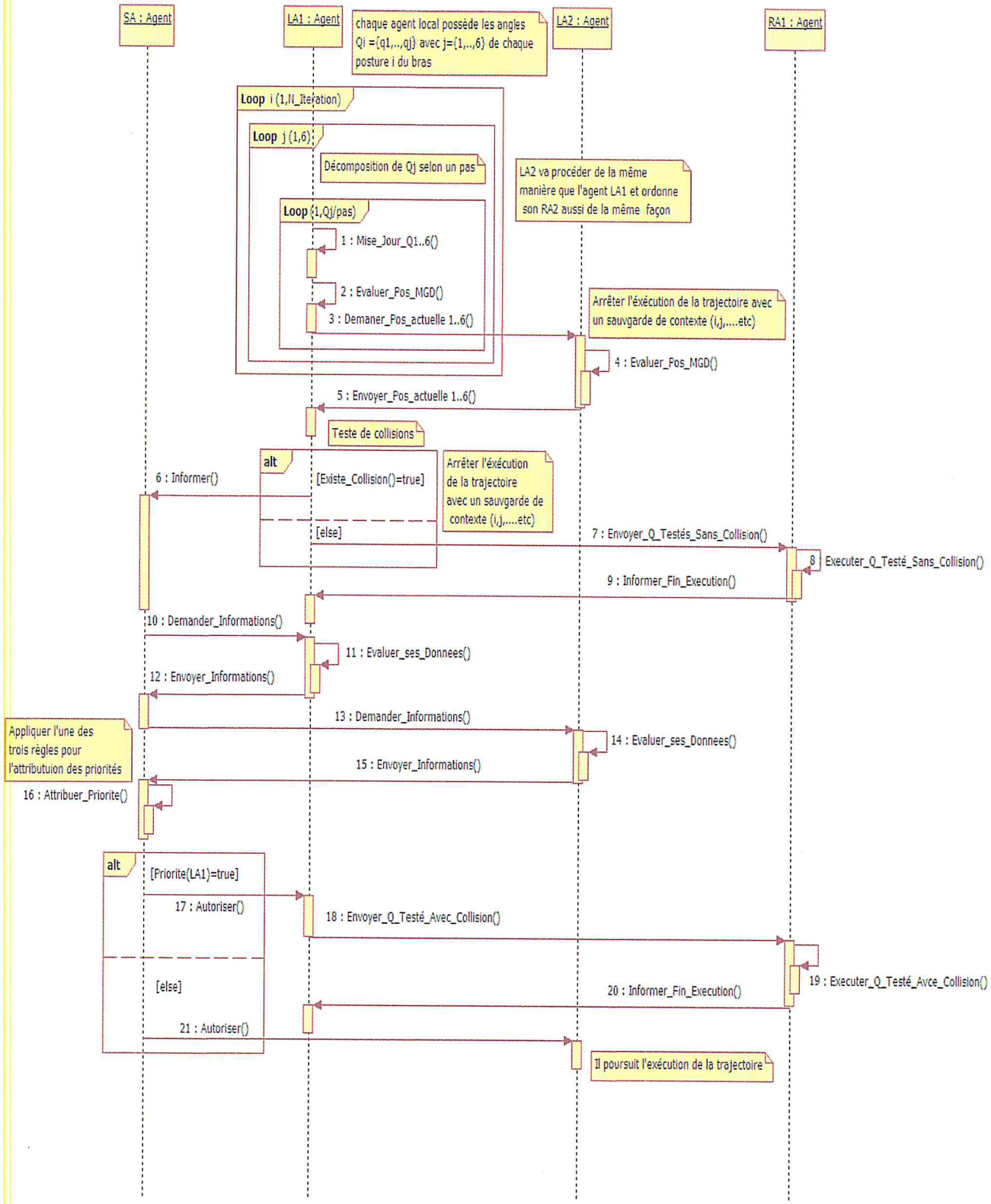


Figure V.5. *Digramme pour La résolution de conflits.*

V.3. Implémentation

Afin d'implémenter notre SMA, nous avons utilisé comme langage de programmation JAVA, en raison de son aspect basé sur la notion d'orienté objet, qui rentre dans l'enjeu du sujet. De plus, la plateforme JADE utilisée, sur laquelle est implémentée notre architecture multi-agent DMACA, s'appuie elle-même sur la notion d'orienté agent, où chaque agent est considéré comme une entité objet. Nous avons aussi utilisé NetBeans comme IDE, et un ensemble d'outils informatiques et robotiques afin d'équiper le terrain de notre travail, nous les verrons par la suite.

V.3.1.JFreeChart

JFreeChart est une API Java permettant de créer des graphiques et des diagrammes de très bonne qualité. Cette API est open source et sous licence LGPL. En revanche, la documentation est payante [18].

JFreeChart fonctionne également avec GNU Classpath, une implémentation libre de la librairie standard en Java.

Exemples de graphiques disponibles sous Jfreechart :

- Graphiques, comme le montre la figure V.6.
- Diagramme camembert
- Diagramme de Gantt
- Histogrammes
- Thermomètres, compas, compteur de vitesse, etc…

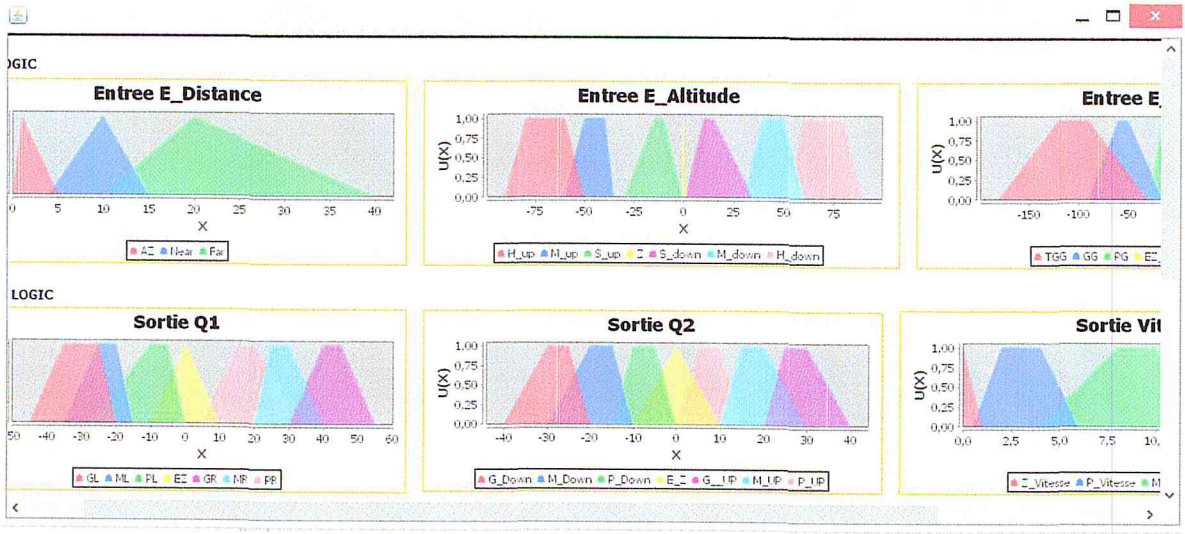


Figure V.6. *Diagrammes des ensembles flous dessinés à l'aide de JFreeChart-1.0.13.*

V.3.2.JFuzzyLogic

JFuzzyLogic est une bibliothèque de la LF, open source écrit en Java et mise en œuvre de normes de l'industrie pour simplifier les développements des systèmes flous. JFuzzyLogic implémente le langage de commande floue (FCL) spécification, ainsi qu'une bibliothèque complète qui simplifie le développement de la LF [19]. La figure V.7 suivante illustre le code implémenté pour notre GTFs.

Robot.fcl.txt

Source | History

```
1  FUNCTION_BLOCK Robot
2  // Define input variables
3  VAR_INPUT
4      E_Distance : REAL;
5      E_Altitude: REAL;
6      E_angle: REAL;
7  END_VAR
8
9  // Define output variable
10 VAR_OUTPUT
11     Q1: REAL;
12     Q2: REAL;
13     Vitesse: REAL;
14 END_VAR
15 // Fuzzify input variable 'E_Distance': {'AZ', 'Near_R' , 'Far_R'}
16 FUZZIFY E_Distance
17         TERM AZ := trian 0 1 5;
18         TERM Near :=trian 4 10 15;
19         TERM Far := trian 10 20 40;
20
21 END_FUZZIFY
22 // Fuzzify input variable 'E_Altitude': {'TGG_', 'GG_' , 'PG_','EZ_','PD_','GD_','TGD_'}
23 FUZZIFY E_Altitude
24         TERM H_UP := trape -90 -80 -60 -50;
25         TERM M_UP := trape -60 -50 -40 -35;
26         TERM S_UP := trape -30 -15 -10 -0.5;
27         TERM Z := trian -1 0 1;
28         TERM S_Down:= trape 0.5 10 15 35;
29         TERM M_Down:= trape 30 40 50 60;
30         TERM H_Down:= trape 50 60 80 90;
```

Figure V.7. *Capture d'écran pour le fichier FCL créé pour le contrôleur flou.*

V.3.3.Plateforme JADE

JADE est une plate-forme multi-agents développée en Java par CSELT (Groupe de recherche de Gruppo Telecom, Italie) qui a comme but la construction des systèmes multi-agents et la réalisation d'applications conformes à la norme FIPA, JADE comprend deux composantes de base : une plate-forme agents compatible FIPA et un paquet logiciel pour le développement des agents Java [20].

Dans la figure V.8, on voit qu'il y a trois rôles principaux dans une plate-forme multi-agents FIPA :

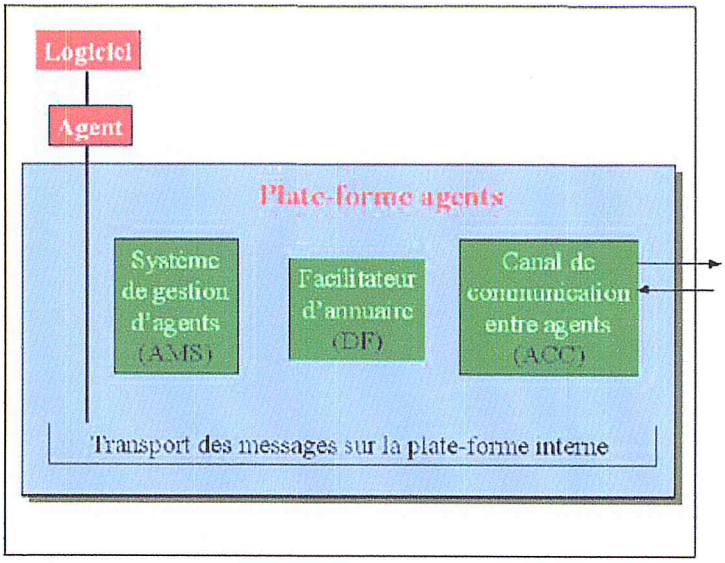


Figure V.8. *Modèle de référence pour une plate-forme multi-agents FIPA* [20].

- **Système de Gestion d'Agents** (Agent Management System - AMS) : est l'agent qui exerce le contrôle de supervision sur l'accès à la plate-forme et son usage; il est responsable de l'authentification des agents résidents et du contrôle d'enregistrements.
- **Canal de Communication entre Agents** (Agent Communication Channel - ACC) : est l'agent qui fournit la route pour les interactions de base entre les agents dans et hors de la plate-forme ; c'est la méthode de communication implicite qui offre un service fiable et précis pour le routage des messages ; il doit aussi être compatible avec le protocole IIOP pour l'interopérabilité entre les différentes plates-formes multi-agents.
- **Facilitateur d'Annuaire** (Directory Facilitator - DF) : est l'agent qui fournit un service de pages jaunes à la plate-forme multi-agents.

V.3.4. Communication entre les agents utilisant JADE

Dans une plateforme multi-agents, il est essentiel que les agents puissent communiquer entre eux pour pouvoir réagir et interagir de manière dynamique. Les agents de l'architecture interagissent entre eux selon des relations sous formes de messages ACL fournis par la plateforme JADE. Ces messages sont composés en général de :

- L'émetteur du message : un champ rempli automatiquement lors de l'envoi d'un message.
- L'ensemble des récepteurs du message : un message peut être envoyé à plusieurs agents simultanément.

- L'acte de communication : qui représente le but de l'envoi du message (informer l'agent récepteur, appel d'offre, réponse à une requête,…)
- Le contenu du message.
- Un ensemble de champs facultatifs, comme la langue utilisée, l'ontologie, le time Out, l'adresse de réponse…

Afin d'assurer la bonne gestion des messages échangés entre les agents de l'architecture DMACA, nous avons utilisé un codage bien défini pour les messages, ce qui nous a permis de bien gérer les données échangées dans les deux sens de communication. Le codage utilise une stratégie simple adaptée à notre système de contrôle, ces messages sont codés comme suit :

Message= Objectif # Paramètre 1#.....# Paramètre N.

Les tableaux suivants (tables V.1 et V.2) résument les différents messages échangés entre les agents du système :

Sender	Receiver	Objectif	Description
LA	LA	« Get_Pos_To_My_Instant»	Pour demander à un agent local les coordonnées cartésiennes de son bras manipulant à un instant donné
LA	LA	« My_Coord»	Pour envoyer ses six coordonnées cartésiennes à l'instant demandé par l'autre agent local
LA	LA	«Fini_Pos»	Pour indiquer à l'autre agent local la fin de l'envoi des coordonnées cartésiennes du bras manipulant pour l'exécution d'une trajectoire
LA	LA	«Collision»	Pour confirmer si la posture dont le récepteur vient d'exécuter actuellement est celle qui a posé la collision avec la posture qui sera exécuté actuellement par l'émetteur
LA	LA	«Conf_Collision»	Réponse de confirmation sur la collision

Table V.1. *Différents types de messages échangés entre les agents locaux.*

Sender	Receiver	Objectif	Description
LA	**SA**	« Presence»	Pour signaler à l'agent superviseur sa présence sur la plateforme JADE
LA	**SA**	« Alone »	Pour indiquer à l'agent superviseur qui est le seul sur la plateforme JADE
LA	**SA**	« Absence »	Pour signaler son absence avant de quitter la plateforme JADE
LA	**SA**	«Priority»	Pour signaler l'existence d'une collision afin d'attribuer les priorités d'exécution de la posture à un instant donné aux agents locaux qui ont posés la collision
LA	**SA**	«My_Infos_Priority»	Pour donner au récepteur les informations nécessaires pour l'attribution des priorités
SA	**LA**	«Authority»	Pour autoriser le récepteur à poursuivre son exécution de trajectoire.
SA	**LA**	« New_List_Agent»	Pour informer l'agent local sur la liste des agents locaux présents pour l'instant sur la plateforme JADE
SA	**LA**	«Infos Priority»	Pour demander au récepteur les informations nécessaires pour le calcul des priorités.

Table V.2 : *Différents types de messages échangés entre l'agent superviseur et les agents locaux.*

Comme indiqué au paravent en ce qui concerne la structure du message, ce dernier est associé en plus de l'objectif, à une liste de paramètres qui peuvent être aussi des entités structurées ayant ses propres encodages selon la nature ou l'objectif du message. Pour l'entête du message, il est automatiquement rempli par les informations (ID_Sender, ID_Reciever, les numéros de port avec les adresses IPs …etc), sous le contrôle de la plateforme JADE dès qu'un agent veut communiquer.

Pour la communication, il y a d'autres types où les deux bornes du fil de communication présentent les agents de la partie esclave. Cette partie, tourne entre LA et son RA, afin de communiquer réellement le robot ou plutôt pour donner des ordres à son bras manipulant, basée sur le principe des communications client/serveur et utilisant comme type de conversation le protocole TCP/IP.

Les messages envoyés par les agents locaux aux agents distants (PCs embarqués) des robots sont sous forme de [34]:

Message : Objectif# Paramètre1# Paramètre2#……..#paramètre n $

Les agents locaux des robots donnent des consignes aux agents distants (pc embarqué des robots) pour les exécuter et d'autre part les distants envoient des rapports d'exécution de leurs tâches. Le tableau 2 décrit les différents messages échangés entre l'agent local et son agent distant associé :

Sender	Receiver	Objectif	Paramètres	Description
LA	RA	« POS »	Q1,…, Q6 °	Pour bouger les axes du bras
LA	RA	« Gripper »	'O'	Ouvrir la pince
LA	RA	« Gripper »	'C'	Fermer la pince
LA	RA	« STOP »	/	Pour arrêter le bras
LA	RA	« CAL »	/	Pour lancer le calibrage du bras
LA	RA	« BRA »	/	Demander les données des capteurs
RA	LA	« End_MOV »	/	Pour informer l'agent local que le mouvement est effectué
RA	LA	« BRA »	Q1…Q6	Les données des capteurs

Table V.3. *Messages échangés entre les agents locaux et les agents distants des robots.*

V.3.5 .Interfaces Homme/Machine développées

Réellement, la structure elle-même de l'agent sous JADE, n'est qu'un objet thread en train de s'exécuter sur la plateforme. Cet environnement va contrôler les conflits existants, et gérer l'ensemble des agents, mais cela ne suffit pas pour contrôler tout le fonctionnement du système. Une interaction homme/machine est nécessaire pour assurer le bon déroulement des mouvements du RM, et afin de bien lancer les tâches de traitement dans le but de satisfaire le système en entier.

Nous nous focalisons à ce stade sur les différentes interfaces homme/machine implémentées pour le contrôle des SMM. On distingue deux types d'IHM, une pour SA « IHM SA » et l'autre pour les agents locaux « IHM LA » qui peut prendre aussi, plusieurs tendances selon le type du manipulateur utilisé.

- **IHM SA :**

Cette interface est composée de quatre parties servant à contrôler les comportements de l'agent superviseur par rapport à son environnement. Ces parties sont échelonnées comme suit

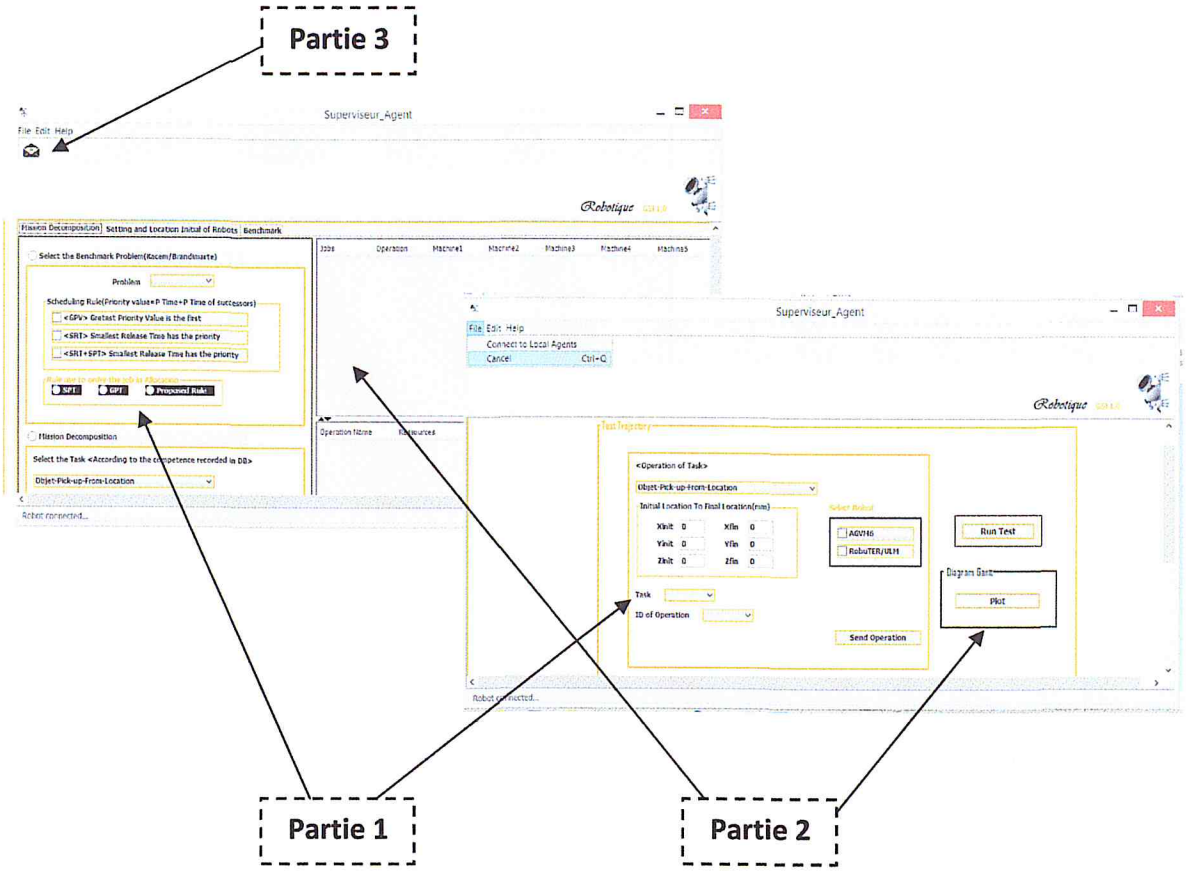


Figure V.9. *Capture d'écran pour l'interface homme/machine SA.*

- **Partie 1 :** cette partie donnée par la figure V.9, permet à l'operateur de définir les opérations d'assemblage pour chaque robot.
- **Partie 2 :** elle permet d'afficher textuellement et graphiquement les données des opérations attribuées aux robots.
- **Partie 3 :** cette partie, sert à visualiser les différents messages échangés entre le robot SA et les agents locaux actifs sur la plateforme.

- **IHM LA :**

Une IHM a été développée pour chaque LA (pour chaque robot) afin de contrôler le robot et de visualiser les données du robot. Elle offre aussi à l'opérateur la possibilité de contrôler le robot manuellement.

L'IHM de chaque robot est présenté dans la figure V.10. Elle est constituée de plusieurs parties :

Figure V.10. *Capture d'écran pour l'interface homme/machine LA.*

- **Partie 1** : Cette partie est pour interagir avec RA, elle permet à l'opérateur de contrôler le robot en mode manuel (Envoyer des configurations, et aussi d'ouvrir ou de fermer la pince).
- **Partie 2 :** Cette Partie sert à l'apprentissage du robot (permet de programmer la trajectoire de robot manuellement), elle permet à l'operateur de définir la trajectoire du robot manuellement. Elle est, elle-même, constituée de deux sous-parties. La première (actionneurs) sert à actionner (envoi des angles au robot) le bras manipulateur en utilisant soit la commande axe par axe, soit en spécifiant les axes à commander.
- **Partie 3** : Pour afficher le plan local de l'agent (les différentes opérations à réaliser).
- **Partie 4** : Le Journal des messages donné par cette partie, afin d'afficher les messages échangés.
- **Partie 5**: cette partie illustre les différentes informations textuelles et graphiques liées aux traitements de l'agent local.

D'après les options données par l'IHM de LA, on trouve que LA présente le cœur dans l'architecture DMACA, puisque il planifie les trajectoires des RMs, et prend en charge tout mouvement exécuté par RA.

V.4. Tests expérimentaux et résultats

Elle comprend les éléments suivants :

V.4.1 .présentation du système expérimental

Dans cette partie, une présentation du système expérimental sera détaillée. Elle donne un aperçu succinct du système mécanique et du système de contrôle bas niveau des robots qu'on va utiliser.

- RobuTER/ULM

L'architecture du RM mobile RobuTER/ULM (figure V.11) est composée d'une plate-forme mobile à commande différentielle sur laquelle est monté un bras manipulateur 6DDL. Le robot est contrôlé par un PC industriel embarqué. et par quatre cartes microcontrôleurs MPC555 communiquant via un bus CAN.

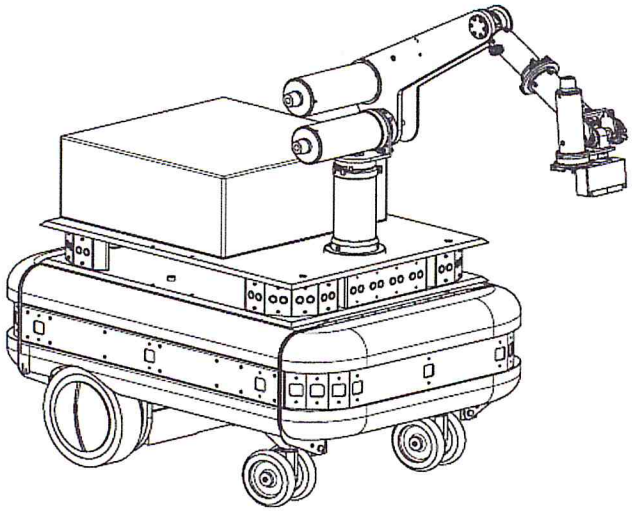


Figure V. 11. *RM mobile RobuTER/ULM.*

- AGVM6

Le Robot AGVM6 (figure V.12) est un bras ultraléger 6DDL. Le robot est contrôlé aussi par un PC embarqué et par deux cartes microcontrôleurs MPC555 communiquant via un bus CAN.

Figure V. 12. *RM AGVM6.*

V.4.2. Scénarios de validation

En raison de non disponibilité du MGD de RM AGVM6, nous avons utilisé le robot RobuTER/ULM dans les tests. Les paramètres ainsi que les limites articulaires du Robot RobuTER/ULM sont donnés par le tableau suivant :

Articulation	V^{max}_i (rad/s)	Q_{min} (°)	Q_{max} (°)
1 (θ_1)	0.4	-95	96
2 (θ_2)	0.4	-24	87
3 (θ_3)	0.4	-2	159
4 (θ_4)	0.4	-50	107
5 (θ_5)	0.4	-73	40
6 (θ_6)	0.4	-91	91

Table V.4. *Paramètres et limites articulaires du Robot RobuTER/ULM.*

- **Scénario de la génération de trajectoire**

Dans ce scénario, nous allons présenter le comportement de RM afin de réaliser une opération introduite par l'utilisateur tout en mettant l'accent sur la génération de trajectoire par le planificateur flou proposé.

L'operateur introduit une opération via l'IHM de SA afin de l'envoyer après au robot RobuTER/ULM pour l'exécution. La figure ci-dessous, présente la partie qui permet d'introduire les opérations et de lancer l'exécution (les envois aux agents locaux pour la PTs, puis le test des éventuelles collisions après l'exécution).

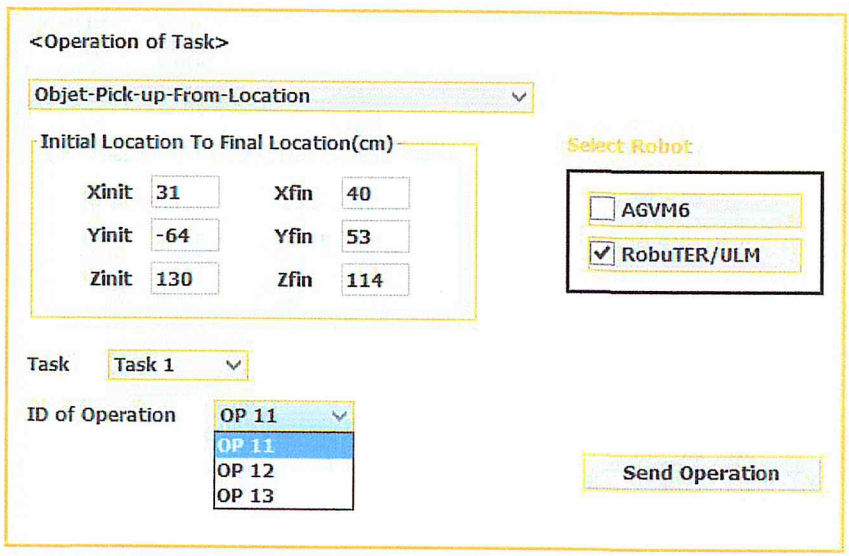


Figure V.13. *Opération attribuée au robot manipulateur RobuTER/ULM.*

Soit un objet à la position P_{init} (31,-64, 130) qui est la position initiale de la pince du bras. L'opération <O_1>consiste à déplacer cet objet à une position finale P_{cible1} (40,53,114)

- La configuration initiale de robot (base, bras) dans le repère de base (toutes les positions sont données par rapport à ce repère) est donnée comme suit :
- Position initiale de la base : $P_{init\_base}(x, y, \varphi)= (0, 0, 0)$ cm.
- Configuration initiale du bras Q_i : Q_{init} $(Q_1,Q_2,Q_3,Q_4,Q_5,Q_6)$= (-60°, 61°,30 °,38°,
- -15°,1°).
- Position initiale de la pince donnée par le MGD (correspondent à Q_{init}) de bras : $P_{init}(X, Y, Z)$= (31,-64,130) cm.

La variation des cordonnées articulaires (profils des trajectoires articulaires) est montrée par la figure V.15. Ainsi que les vitesses des articulations dans la figure V.14.

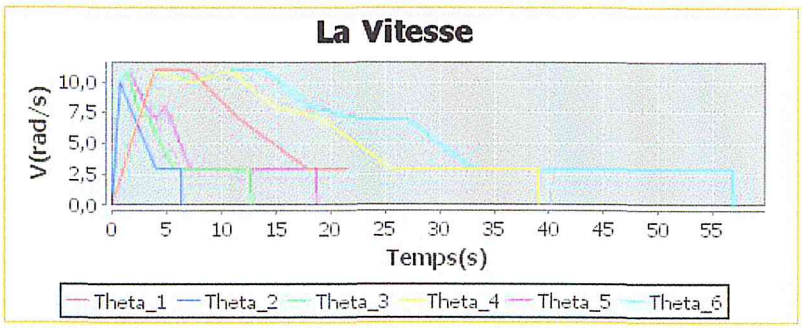


Figure V.14. *Vitesses des angles articulaires pour la trajectoire de l'opération <O_1>.*

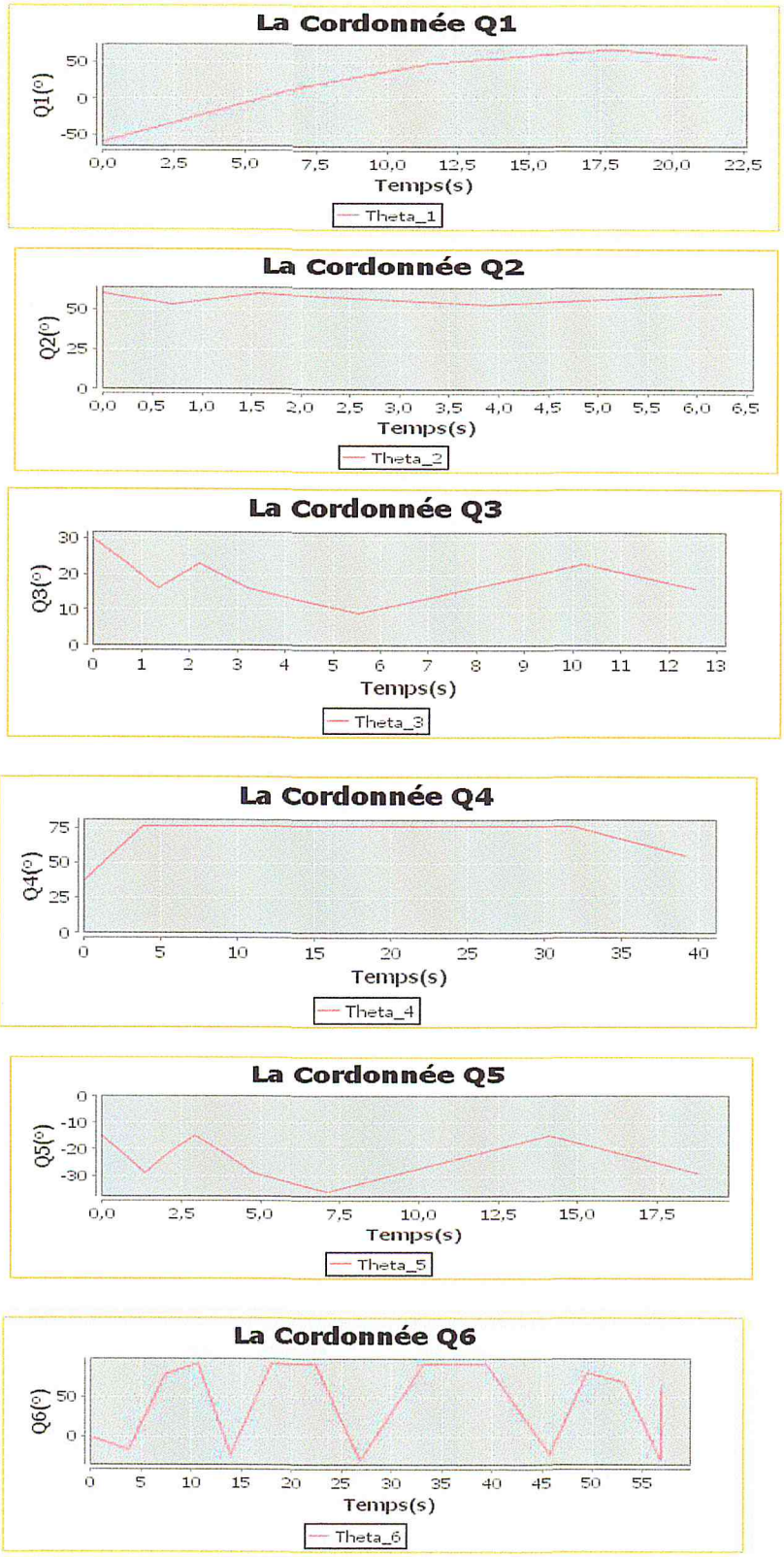


Figure V.15. *Angles articulaires du bras manipulant générés pour la trajectoire de l'opération <O1>.*

N_itération	Angle articulaire	Configuration articulaire	Position de l'effecteur	Type Q°	Q° généré	Vitesse (rad/s)
0	/	(-60, 61, 30, 38,-15,1)	(31, -64, 130)	/	/	/
1	Q1	(-18 61 30 77 -15 43)	(67, -26, 133)	RL	42	11
2	Q2	(-18 54 23 77 -22 43)	(70, -24, 117)	UP/Down	-7	10
3	Q3	(-18 54 16 77 -29 43)	(67, -21, 110)	UP/Down	-7	11
4	Q4	(-18 54 16 77 -29 79)	(67, -21, 110)	RL	36	10
5	Q5	(-18 54 16 77 -22 79)	(68, -24, 110)	UP/Down	7	10
6	Q6	(-18 54 16 77 -22 91)	(68, -24, 110)	RL	36	11

Table V.5. *Échantillon d'une trajectoire générée pour l'opération* $<O_1>$.

La méthode proposée est inspirée du raisonnement humain lors de la génération d'une trajectoire. Elle a été testée pour plusieurs opérations, et parmi ces tests, les résultats présentés dans les deux figures (V.13 et V.14) de l'opération décrite précédemment. On constate bien, d'après les sorties obtenues par le générateur flou (les angles articulaires associés avec leurs vitesses), que les articulations du bras manipulateur atteignent la position finale de l'opération en des temps raisonnables, ainsi les butées articulaires, et les limites des vitesses sont respectées pour la trajectoire générée.

L'échantillon de la trajectoire présenté dans la table V.5, décrit les contraintes de dépendances entre les angles, dont chaque angle Q influe sur les angles successeurs possédants son type. En effet, si on prend l'exemple de la première itération, on constate que lorsque Q1 s'accroît par 42°, les angles Q4 et Q6s'accrois de la même valeur. Les variations de ses vitesses viennent pour assurer cette contrainte, par la comparaison entre les graphes de ces derniers, pendant l'intervalle du temps [0,4] présentée par la figure V.14 ci-dessus.

V.5. Conclusion

Ce chapitre a présenté les stratégies implémentées après une étude conceptuelle et fonctionnelle, pour résoudre le problème de planification de trajectoires de deux robots manipulateurs se situant dans le même espace de travail.

La méthode proposée dans le cadre de planification de trajectoires sans collisions entre robots manipulateurs a ses caractéristiques, ses avantages et ses inconvénients. Cependant, le but final est de trouver les meilleures règles d'inférences en plus de l'ensemble des règles admissibles dans le contrôleur flou proposé, afin de générer des trajectoires atteignables tout en minimisant la présence du problème de minimas locaux.

Conclusion et perspectives

Les travaux présentés dans ce manuscrit portent sur la planification des trajectoires sans collisions pour un SMR. Notamment pour des RMs de 6DDL. Nous nous sommes consacrés au début à la présentation des différentes notions de base liées au domaine robotique, et les SMR. Nous avons également présenté l'architecture de contrôle multi-agents pour notre système. Ensuite, nous avons passé en revue les différentes approches existantes pour la PTs, suivies d'une étude comparative des travaux réalisés dans ce domaine.

Ainsi, les méthodes proposées dans ce sujet, consistent à créer un GFTs, les trajectoires générées par ce dernier, seront contrôlées pendant l'exécution, afin d'éviter les collisions qui peuvent exister au sien d'une flotte de RMs.

Le générateur proposé permet de planifier les trajectoires du RM RobuTER/ULM, utilisé avec succès. Cependant, cette technique dédiée à la génération des trajectoires porte quelques situations d'échec, présentant le problème de minimas locaux. Cela, est dû au nombre de degré de liberté qui caractérise un bras manipulant, ce qui rend le calcul d'une trajectoire difficile avec des outils basés sur la LF.

De plus, le manque du matériel pour tester l'approche proposée pour l'évitement de collions, était un véritable obstacle pour nous, surtout pour nous aider à améliorer la technique par des tests réels sur un ensemble de robots.Toutefois, les résultats obtenus par l'application implémentée sur l'architecture multi-agents, ont aussi donné de bons résultats dans l'espace théorique pour ce problème.

Afin d'améliorer le travail réalisé, les perspectives suggérées de ce travail peuvent se résumer dans les points suivants :

- L'amélioration du jeu des règles d'inférences du générateur flou de trajectoires créé, par l'ajout de règles obtenues par les tests expérimentaux.
- L'utilisation d'une technique permettant au robot de sortir d'un minima local, de telle sorte qu'il pourra poursuivre sa planification de trajectoire.
- Afin de respecter les contraintes temporelles et fonctionnelles décrites par l'espace de travail d'un robot, une amélioration des règles d'attribution de priorités pour l'évitement de collisions est nécessaire.

Webographie

[1] Site officiel **«notrefamille» :**

http://www.notrefamille.com/dictionnaire/definition/robotique (dernière visite le 25/2/2016).

[2] Site officiel **«futura-sciences» :**

http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/robotique-robotique-603/ (dernière visite le 25/2/2016).

[3] Site officiel **«larousse» :**

http://www.larousse.fr/dictionnaires/francais/automate/6746 (dernière visite le 25/2/2016).

[4] Site officiel **«futura-sciences» :**

http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/robotique-automate-848/ (dernière visite le 25/2/2016).

[5] Site officiel **«perso.univ-lyon2» :**

http://perso.univ-lyon2.fr/~mollon/EO-web/12-13/Virgo/definition%20et%20historique%20de%20la%20robotique/definition%20et%20historique%20de%20la%20robotique.html (dernière visite le 25/2/2016).

[6] Site officiel **«cinematheque» :**

http://www.cinematheque.fr/objet/347.html (dernière visite le 01/06/2016).

[7] Site officiel **«lesrobotsfautilenavoirpeur.weebly» :**

http://lesrobotsfautilenavoirpeur.weebly.com/la-robotique-spatiale.html (dernière visite le 01/06/2016).

[8] Site officiel **«jf.duchet.pagesperso-orange» :**

http://jf.duchet.pagesperso-orange.fr/robotique/Pages/applications.htm (dernière visite le 01/06/2016).

[9] Site officiel **«medtroniceureka» :**

http://www.medtroniceureka.com/fr/innovation-articles/inspiration/contrib_156176

(dernière visite le 01/06/2016).

[10] Site officiel **«telegraph» :**

http://www.telegraph.co.uk/motoring/news/8876686/Honda-Asimo-robot-advances-in-leaps-and-bounds.html (dernière visite le 01/06/2016).

[11] Site officiel **«notre-planete» :**

http://www.notre-planete.info/actualites/4095-robots-agriculture (dernière visite le 01/06/2016).

[12] Site officiel **«realagriculture» :**

https://www.realagriculture.com/2011/03/london-farm-show-halex-gt-is-apparently-not-just-a-fancy-weed-picking-robot/ (dernière visite le 01/06/2016).

[13] Site officiel **«soocurious» :**

http://soocurious.com/fr/science-robot-interaction-robot-drone-decouverte/ (dernière visite le 01/06/2016).

[14] Site officiel **«blog.schneider-electric» :**

http://blog.schneider-electric.fr/industrie/2014/07/24/developpement-robots-au-service-du-monde-industriel/ (dernière visite le 01/06/2016).

[15] Site officiel **«humanoides» :**

https://humanoides.fr/wpcontent/uploads/2013/04/Humanoides_fr_robot_industriel_cover.jpg (dernière visite le 25/2/2016).

[16] Site officiel **«inra» :**

https://www7.inra.fr/mia/M/fispro/fisprodocfr/aide-enligne/node39.html(dernière visite le 2/6/2016).

[17] Site officiel **«ferdinandpiette» :**

http://www.ferdinandpiette.com/blog/2011/08/exemple-de-systeme-flou-un-planificateur-de-trajectoire/ (dernière visite le 2/6/2016).

[18] Site officiel **«lrie.efrei» :**

http://lrie.efrei.fr/2009/11/jfreechart-creer-des-graphes-et-diagrammes-en-java/ (dernière visite le 2/6/2016).

[19] Site officiel **«jfuzzylogic.sourceforge»:**

http://jfuzzylogic.sourceforge.net/html/manual.html (dernière visite le 2/6/2016).

[20] Site officiel **« turing.cs.pub.ro » :**

http://turing.cs.pub.ro/auf2/html/chapters/chapter6/chapter_6_5_1.html (dernière visite le 2/6/2016).

Bibliographie

[21] Chitic. Stefan-Gabriel, Ponge, Julien, et Simonin, Olivier. *Intergiciels pour systèmes multi-robots* : état de l'art. In : *UbiMob2014: 10èmes journées francophones Mobilité et Ubiquité*, 2014. p. 8 p.

[22] Y. Zhi. *Contributions à la coordination de tâches et de mouvements pour un système multi-robots* : thèse de doctorat, Spécialité : Informatique, Université Paris 8, France, 2012.

[23] J.Ferber. *Les Systèmes Multi Agents: vers une intelligence Collective*, 1997.

[24] L. Soltani. *Les Systèmes Multi-Agent pour le Contrôle de Production* : thèse de doctorat, Spécialité : Génie Industriel, Université HADJ LAKHDAR BATNA, Algérie, 2007.

[25] R. Mandiau, Emmanuelle Grisling Lestrugeon. *Systèmes Multi-agents*. Techniques de l'ingénieur, traité Informatique Industrielle S7216. 2002.

[26] S. Labidi, W. Lejouad. *De l'Intelligence Artificielle Distribuée aux Systèmes Multi-Agents*, Rapport de recherche n°2004, 39 pages, 1993.

[27] Ferguson Innes (1992a). *Touring Machines: Autonomous Agents with Attitudes*. In IEEE Computer, 25 May.

[28] C. Berchet. *Modélisation pour la simulation d'un système d'aide au pilotage industriel* : thèse de doctorat, Spécialité : Génie Industriel, institut National Polytechnique de Grenoble, France, 2000.

[29] A .Maoudj et al. *Multi-Agent Architecture for Telerobotics of Heterogeneous Multi-robot Cooperation through Negotiated Task Allocation with Constraints*, Division Productique et Robotique, Centre de Développement des Technologies Avancées, BP 17, Baba Hassen, Alger 16303, Algérie

[30] A. Hentout, *Architecture Multi-agents Dédiée au Contrôle des Manipulateurs Mobiles : Application au Robot RobuTER/ULM* : thèse de doctorat, Université des Sciences et technologies Houari Boumediene, Algérie, 2012.

[31] P. Bonasso, D. Kortenkamp, D. Miller, M. Slack. *Experiences with an Architecture for Intelligent, Reactive Agents*. Journal of Artificial Intelligence Research, 9:1, 1997.

[32] W. Khalil, E. Dombre, Hermes Penton *Modeling, Identification & Control of Robots*, Science 2002, 480 pages .

[33] X. Broquere. *Planification de trajectoire pour la manipulation d'objets et l'interaction Homme-robot* : thèse de doctorat, Spécialité : Informatique et Robotique, Université Toulouse III Paul Sabatier (UT3 Paul Sabatier), France, 2011.

[34] I. Akli. *Coordination de Mouvements en télé intervention de robots manipulateurs mobiles* : thèse de doctorat, Spécialité : Electronique, Université des Sciences et de la Technologie Houari Boumediene, Algérie, 2015.

[35] A. Hentout. *Planification de trajectoires sans collisions pour robots manipulateurs* : PFE de Magister, Spécialité : Informatique Industrielle, Ecole Militaire Polytechnique, Algérie, 2003.

[36] H. Ghodbane, M. Moussaoui, O. Kazar. *Commande d'un bras manipulateur 6 ddl évitement d'obstacle par la logique floue*. Courrier du Savoir, 2010.

[37] S. Kermiche. *Modélisation et commande d'un robot par méthodes intelligentes* : thèse de doctorat, Spécialité : automatique industrielle, Université BADJI MOKHTAR-ANNABA, Algérie, 2006.

[38] D. Mokadem. *Contrôle Flou des Processus Biotechnologiques à Base d'Algorithmes Génétiques* : thèse de doctorat, Spécialité : Electronique, Université FERHAT ABBAS DE SETIF, Algérie, 2010.

[39] M. Beggas. *Modélisation par un système multi-agents d'un hypermédia éducatif adaptatif dynamique* : PFE de Magister, Spécialité : Informatique, Centre Universitaire d'Eloued, Algérie, 2005.

[40] David Popenoe. Sociology (11th Edition). Prentice Hall, 1999.

[41] Y. Uny Cao, Alex S. Fukunaga, and Andrew B. Kahng. Cooperative mobile robotics : Antecedents and directions. Autonomous Robots, 4(1) :7–27,1997.

[42] T. Lozano-Pérez. Spatial Planning : A Configuration Space Approach. IEEE Transactions on Computers, vol. C–32, no. 2, pages 108–120, Feb. 1983.

[43] Versino .C, and Gambardella, L. M. "learning fine motion by using the hierarchical extended Kohonen Map". In Von der Malsburg, C, Von Seelen, W. Vorburggen, J. C;, and Sendroft, B. (Eds). Proc. ICANN96, International conference on Artificial Neural Networks, Bochum, Germany, 17-19 July, Vol. 1112 of lecture Notes in Computer Science , Springer-Verlag, Berlin, pp 221-226, 1996.

[44] Versino, C. and Gambardella, L. M "learning fine motion in Robotics: Design and Experiments", IDSIA, Corso Elvesia 36, 6900 lugano, Switzerland, by CRC press LLC, 2000.

[45] Heikkonen, J., Koikkalainen, P., and Oja, E. "Motion behavior learning by self-organization". Proc. ICANN93, International on Artificial Neural Networks, Amsterdam, The Netherlands, September 13-16, pp 262-267, 1993.

[46] Knobbe, A. J., Kok, J. N., and Overmars, M. H, " Robot Motion Planning in Unknown environments using neural networks". Proc . ICANN95, International Conference on Artificial Neural Networks, Paris, France, October 9-13, pp 375-380, 1995.

[47] Fritzke, B. " A growing neural gas network learns topologies". Advances in Neural information Processing Systems 7, Tesauro, G., Touretzky, D.S., and Leen T. K . (Eds.), MIT, Press, Combridge , MA, pp 625-632, 1995.

[48] Heikkonen, J. Millan, J. del R. and Cuesta , E. " Increamental learning from basic reflexes in an autonomous mobile robot". Proc. EANN93, International Conference on Enginnering Applications of Neural Networks, Otaniemi, Espoo, Finland , August 21-23, pp 119-126, 1995.

[49] Millan, J. del R. " Reinforcement learning of goal-direct obstacle-avoiding reaction strategies in an autonomous mobile robot". Robotics and Autonomous systems, vol. 15, no.3, pp 275-299, 1995.