UNIVERSITÉ DE BLIDA -1-

Faculté des Sciences

Département d'Informatique

MÉMOIRE DE MAGISTER

Spécialité : Informatique Répartie et Mobile (IRM)

Développement d'un contrôleur multi-agents pour les robots manipulateurs mobiles : Cas du robot RobuTER/ULM

Par

MESSOUS Mohamed Ayoub

Devant le jury composé de :

Mme. N. BENBLIDIA	Maître de Conférences(A), U. de Blida	Présidente
M. F. FERGUENE	Maître de Conférences(A), USTHB, Alger	Examinateur
Mme. N. BOUSTIA	Maître de Conférences(A), U. de Blida	Examinatrice
M. A. HENTOUT	Maître de recherche(B), CDTA, Alger	Co-promoteur
Mme. S. OUKID	Maître de conférences(A), U. de Blida	Promotrice

Blida, Mars 2014

RÉSUMÉ

Le but de ce travail est de développer une approche de contrôle multi-agents pour les robots manipulateurs mobiles autonomes. Le schéma de contrôle proposé assigne un agent réactif (Agent axe) pour le contrôle de chaque articulation du manipulateur, un agent hydride (Agent base) pour le contrôle de la base mobile, ainsi qu'un agent hybride (Agent superviseur) pour assurer la coordination et la synchronisation entre tous les autres agents du système. Chaque agent de contrôle possède, indépendamment des autres agents, un objectif local qui est de minimiser la distance entre la position de l'effecteur du manipulateur et celle de la cible imposée. L'avantage principal pour une telle approche est l'assurance d'une certaine tolérance aux pannes et aux anomalies qui peuvent surgir, sans envisager un mécanisme spécifique.

Les résultats de simulation font appel à de différents scenarios de validation, sans et avec considération des défaillances de quelques articulations du manipulateur ou de la panne de la base mobile. Des résultats satisfaisants ont été obtenus pour toutes les tâches considérées.

Mots clés : Manipulateurs mobiles, Approches de contrôle, Système multi-agents, RobuTER/ULM.

ABSTRACT

The purpose of this work is to design a multi-agent approach for controlling autonomous mobile manipulators. The proposed scheme assigns a reactive agent (Joint agent) to control each articulation of the manipulator, a hybrid agent (Mobile base agent) for the control of the mobile base, and a Supervisory agent to assure coordination and to synchronize the work of the whole of the precedent agents. Each control agent has its own local goal to be reached independently from the other agents. This local goal consists of bringing the position of the end-effector as close as possible to the imposed target. The main advantage of such an approach is that the system pledges a fault-tolerant response to certain types of breakdowns without adding any specific dysfunction treatment.

The simulation results are given via different operational positioning scenarios with and without considering breakdowns of some articulations of the manipulator or the mobile base. The proposed approach revealed satisfying results for all the considered tasks.

Keywords: Mobile Manipulators, Control approaches, Multi-agent Systems, RobuTER/ULM.

ملخص

الهدف من هذا البحث هو تطوير منهج للتحكم متعدد الوكلاء من اجل ذراع الي متحرك. مخطط التحكم المقترح ينسب وكيل ذو سلوك تفاعلي (وكيل–مفصل) للتحكم بمفصل الذراع الالي، وكيل هجين (وكيل–قاعدة) للتحكم في الروبوت المتحرك، بالإضافة الى وكيل هجين (وكيل–مراقب) لتحقيق الترابط و التزامن بين كل الوكلاء الأخرين. كل وكيل تحكم لديه، و بشكل مستقل عن باقي الوكلاء، هدف خاص به. هذا الأخير يتمثل في تقليل المسافة الى الحد الأقصى بين موضع نهاية الذراع الآلي وموضع الهدف المنشود. الفائدة الرئيسية من هذا المنهج هو ضمان التسامح مع الأعطاب و الحالات الشاذة التي تبرز على نحو غير متوقع من دون تصور آلية خاصة لهذا الغرض. نتائج المحاكاة تتضمن نتائج انجاز سيناريوهات مختلفة لعدة مهام مع وبدون اعتبار اعطاب على مستوى بعض مفاصل الذراع الالي او القاعدة المتحركة. النتائج المتحصل عليها كانت مرضية لكل المهام المعالجة.

كلمات مفتاحية: ذراع الي متحرك، مناهج التحكم، نظام متعدد الوكلاء، RobuTER/ULM.

DÉDICACES

À mes parents…

REMERCIEMENTS

Mes remerciements sincères et profonds s'adressent à M. HENOUT Abdelfetah, Maître de recherche au CDTA, mon promoteur et mon chef de projet, de m'avoir proposé ce sujet de Magister. Qu'il trouve ici l'expression de ma parfaite gratitude, pour l'intérêt qu'il a porté à mon travail, pour ses conseils et orientations, et surtout pour son support.

Je tiens à exprimer ma reconnaissance à Mme. OUKID Saliha, Maître de conférences à l'USDB, d'avoir acceptée de codiriger ce travail et pour l'aide indéniable qu'elle m'a apporté.

Je souhaite remercier profondément Mme. BENBLIDIA Nadjia, Maître de conférences à l'USDB, qui m'a fait l'honneur de présider mon jury de soutenance. Je tiens également à lui présenter ma parfaite considération pour son dévouement et sa disponibilité.

J'exprime mes vifs remerciements à Mme. BOUSTIA Narhimene, Maître de conférences à l'USDB, et à M. FERGUENE Farid, Maître de conférences à l'USTHB, d'avoir acceptés de juger mon travail.

Que ATTAF Sarah, AKLI Isma, TCHENDERLI Smain et YACEF Fouad trouvent ici l'expression de mes sincères remerciements pour leurs inestimable soutien et leurs précieux remarques et suggestions.

Je remercie également tous mes amis et mes collègues pour leurs encouragements, leurs aides et leur support.

LISTE DES FIGURES

LISTE DES TABLEAUX

TABLE DES MATIÈRES

INTRODUCTION GÉNÉRALE

Nombreux sont les laboratoires de recherche à travers le monde qui se sont investis dans les aspects pluridisciplinaires de la robotique, notamment dans le domaine industriel, militaire et la robotique de service. Les applications classiques relèvent uniquement de la manipulation (systèmes articulés) ou de la locomotion (véhicules robotisés). Ainsi, dans la littérature, nous distinguons principalement deux catégories de travaux :

- ceux utilisant des bras manipulateurs, fixés sur des socles rigides, pour des applications robotiques comme dans le domaine manufacturier où ces robots interviennent dans les chaînes de production pour accomplir des tâches de manipulation, de manutention, d'assemblage, etc. à l'aide des pinces ou autres effecteurs.
- d'autres travaux utilisant des plateformes mobiles ayant la capacité de se déplacer dans leur environnements grâce à des outils de locomotions (roues, pattes, etc.) pour des applications telles que le transport d'objets, l'inspection et le nettoyage des sites à risques, etc.

Cependant, les exigences actuelles nous mènent vers des applications nécessitant, à la fois, la manipulation et la locomotion. Dans ce contexte, les manipulateurs mobiles sont les plus aptes à accomplir ces applications, vu leurs capacités sur le plan locomotion et manipulation. La manipulation est offerte par le bras manipulateur, alors que la locomotion est assurée par la base mobile. Les manipulateurs mobiles acquièrent, ainsi, une grande mobilité ce qui permet d'étendre,

par conséquent, leur espace de travail et d'incrémenter leur capacités opérationnelles [51]. De ce fait, ils peuvent accomplir la plupart des tâches usuelles de la robotique nécessitant, en même temps, des capacités de locomotion et de manipulation. Ces robots ont des applications dans plusieurs domaines :

- le domaine manufacturier pour accomplir des tâches de saisie et de transport d'objets, ou pour accomplir des tâches de manutention.
- le domaine militaire pour libérer l'homme des tâches dangereuses tels que le déminage, la décontamination, etc.
- etc.

Ces robots sont constitués de deux sous-systèmes mécaniques hétérogènes. Ainsi, leur utilisation implique divers problèmes. Nous citons, en particulier, le développement d'une architecture de contrôle, la modélisation de l'environnement (généralement non structuré, inconnu ou partiellement connu), la perception de l'environnement et la fusion multi-sensorielle, la commande unifiée du système robotique, la planification de trajectoires, etc.

Un manipulateur mobile doit être capable d'effectuer les tâches planifiées dans des environnements complexes, inconnus et dynamiques en utilisant seulement ses ressources physiques et informatiques limitées avec une intervention humaine réduite. Par conséquent, un manipulateur mobile doit être doté d'une certaine autonomie. Une tolérance aux pannes et une sureté de fonctionnement sont également des critères de rigueur pour la réussite de ces tâches. Il y a lieu de signaler que même les robots des dernières générations ne peuvent encore exécuter que des tâches prédéfinies avec une intelligence réduite, et la plupart d'entre eux conviennent à une situation et un usage bien définis [30]. De ce fait, plusieurs chercheurs en robotique se sont confrontés au problème de construire une architecture de contrôle.

Une architecture de contrôle est un ensemble organisé de fonctions élémentaires que le robot peut réaliser. Le contrôleur sera l'entité qui manipulera cette architecture pour réaliser les objectifs assignés par l'opérateur. Une architecture de contrôle doit être générique et complète, c.-à-d. elle doit être indépendante de l'architecture matérielle, des différents équipements et ressources existants/opérationnels et logiciels du robot à contrôler. Elle permet, à la fois, de répondre à des exigences de niveau bas (commande des actionneurs, etc.) et aux exigences de niveau haut (tolérance aux fautes, etc.) [19].

Une approche de contrôle est considérée comme étant un ensemble de modèles et techniques utilisés pour réussir le processus de contrôle de robots. Elle est définie comme la méthodologie adoptée lors du traitement des données en entrée, issues des différents capteurs du robot, pour offrir, en sortie, les consignes de contrôle.

Différentes approches de contrôle des manipulateurs mobiles ont été proposées dans la littérature. Elles peuvent être divisées, principalement, en deux grandes classes :

- Les approches traditionnelles/classiques sont basées sur l'étude des modèles mathématiques des structures mécaniques constituant le robot. Le contrôle consiste, alors, à calculer le mouvement des articulations du manipulateur et celui de la base mobile. Pour cela, l'étude des modèles géométriques direct (MGD) et inverse (MGI) est nécessaire. Ces approches produisent des résultats précis, et offrent un contrôle exact pour les tâches répétitives dans des environnements contrôlés et connus (industriels, etc.). Il faut noter que le calcul du MGD suit des règles génériques, alors que le MGI est construit suivant la structure mécanique du robot. De ce fait, le calcul du MGI ne tolère pas de changements dans la structure mécanique du robot (défaillance d'une articulation, de la base mobile, etc.) sans l'ajout de modes spécifiques de traitement de ces pannes. Enfin, les approches classiques ont un autre inconvénient qui est le temps de calcul très important en fonction du nombre de degrés de liberté du robot, en particulier, dans des environnements dynamiques ou inconnus où opèrent la plupart de ces robots.
- Les approches multi-agents proposent une décomposition du contrôle du robot entre un ensemble d'agents distincts ayant des fonctionnalités dispersées. Les agents du système disposent d'une certaine intelligence et sont capables de traiter certaines opérations de manière indépendante, tout en améliorant les capacités interactives et synergiques du système global. Dans ces approches, chaque agent essaie d'aligner la position actuelle de l'effecteur avec celle de la cible, sans connaissance préalable des actions des autres agents. En agissant de manière récursive, les autres agents tentent de faire la même opération. Un comportement global peut émerger, par conséquent, de tous ces agents locaux afin de satisfaire l'objectif global (atteindre la cible). Ces approches offrent des solutions simples et bénéficient des avantages de la résolution distribuée des problèmes. En outre, ces approches ne nécessitent pas de modèles mathématiques et ni de résolution des équations différentielles. Par conséquent, il y a une diminution considérable de l'effort de conception et de temps de calcul par rapport aux approches classiques. Aussi, le fait d'utiliser la technologie multi-agents peut réduire considérablement les difficultés du système en matière de contrôle et d'implémentation et peut améliorer la stabilité et la capacité de coordonner le fonctionnement du système [**55**]. Enfin, les approches multi-agents sont plus souples à être appliquées à n'importe quel robot (mobile, manipulateur et manipulateur mobile).

Le travail de recherche effectué ici s'insère dans la deuxième classe d'approches de contrôle de robots autonomes. Il y a lieu de signaler que la plupart des chercheurs de la littérature, développant des architectures multi-agents de contrôle des manipulateurs mobiles, testent les performances de leurs approches sur des cas simples de robots, évoluant dans des environnements à deux dimensions et accomplissant des tâches simples de délivrance d'objets. Malheureusement, très peu de travaux ont été réalisés en trois dimensions. À cet effet, l'objectif principal de ce travail consiste à développer un contrôleur multi-agents pour les manipulateurs mobiles ayant un seul manipulateur et évoluant dans un environnement à trois dimensions. La tâche de validation consiste à atteindre une situation finale imposée à son effecteur $Effector_{Fin}(x_{EFin}, y_{EFin}, z_{EFin})$ afin de délivrer un objet. De ce fait, les deux ressources hétérogènes, en l'occurrence le manipulateur et la base mobile, sont sollicitées. La tâche a été exécutée sans et avec considération des défaillances de quelques articulations du manipulateur et sans et avec considération de la panne de la base mobile.

Nous avons tenté de trouver un compromis entre faisabilité, viabilité et simplicité de réalisation, en se basant sur les techniques de l'intelligence artificielle distribuée. L'essentiel de nos efforts porte, principalement, sur le développement des aspects décisionnels, architecturaux et interactionnels du système, permettant la maitrise des éventuels incidents et évènements inattendus.

Pour assurer une meilleure présentation du travail effectué et garantir la clarté du mémoire, outre cette introduction générale, ce manuscrit se compose de quatre chapitres, une conclusion générale et deux annexes. Chacun met en évidence une contribution particulière du travail :

- Dans le premier chapitre de ce mémoire, nous présentons un état de l'art des manipulateurs mobiles, leurs domaines d'application, ainsi que leurs principales caractéristiques.
- Dans le second chapitre, nous décrivons les différentes architectures et approches de contrôle des manipulateurs mobiles existantes, ainsi que les travaux de recherche portant sur le développement des approches de contrôle dédiées à ce type de robots, dont les approches basées sur les systèmes multi-agents.
- Dans le troisième chapitre, nous proposant notre approche de contrôle basée sur les systèmes multi-agents. Une analyse du système robotique expérimentale, en occurrence le manipulateur mobile RobuTER/ULM, est présentée en premier lieu. En second lieu, nous décrivons l'architecture globale de l'approche proposée, les détails de la modélisation, la spécification, la composition ainsi que l'échange d'informations inter-agents en ayant recours aux diagrammes UML. Enfin, nous détaillons la mise en œuvre de l'approche proposée.
- Dans le quatrième chapitre, nous testons l'approche proposée via différents scénarii de validation. Pour cela, les résultats de simulation sont présentés, ainsi qu'une étude comparative avec les approches citées dans la littérature.

- Une conclusion générale, donnée à la fin du manuscrit, résume la contribution du travail effectué, discute les principaux résultats obtenus et fixe des perspectives envisageables d'amélioration.

Le manuscrit compte aussi deux annexes données comme suit :

- L'annexe A illustre les principales étapes de calcul du modèle géométrique direct *RobuTER/ULM*.
- L'annexe B décrit la structure global du fichier trace résultat de la simulation des différents scénarios de validation de l'approche proposée.

CHAPITRE 1
MANIPULATEURS MOBILES

1.1. Introduction

Un système automatisé est une machine ou un dispositif qui fonctionne de façon automatique ou en réponse à une commande à distance. Le terme « robot » nous vient du mot tchèque « *robota* » qui signifie travailleur compulsif. Bien que l'image d'un androïde ou d'une quelconque machine ayant une forme humaine nous vienne à l'esprit lorsque nous parlons de robots, la définition de ce terme s'applique aussi bien aux systèmes automatisés tels que grille-pain automatiques et cuisinières électriques.

Un manipulateur est composé d'une structure articulée (plusieurs corps rigides assemblés par des liaisons), fixée sur un socle rigide. Ce type de robots est utilisé principalement pour libérer les opérateurs de tâches lassantes, monotones et répétitives. Ce qui rend l'utilisation des manipulateurs très répondu dans le domaine manufacturier, où ils peuvent intervenir, dans les chaînes de production, pour accomplir des tâches de manutention, d'assemblage, etc. Toutefois, la

morphologie des manipulateurs à base fixe, et les dimensions limitées des corps qui les constituent, créent des espaces de travail[1] très limités.

En fixant un manipulateur sur une base mobile, c-à-d. combiner la locomotion à la manipulation, on obtient ce qu'on appelle un *manipulateur mobile*. La mobilité, offerte par la base mobile, permet au manipulateur d'étendre considérablement son espace de travail atteignable et d'incrémenter ses capacités opérationnelles [51]. Le nouveau système serait donc capable d'assurer des tâches de manutention et d'intervention qui nécessitent à la fois la mobilité de la base ainsi que la manutention du manipulateur. La figure suivante représente le manipulateur mobile *MM-500*.

En robotique, le concept d'un manipulateur mobile est très simple. Il s'agit de réunir, dans un même système, un (des) moyen(s) de locomotion à un (des) moyen(s) de manipulation. Le(s) moyen(s) de locomotion assure(nt) au système un espace de travail limité principalement par son autonomie énergétique. Le(s) moyen(s) de manipulation assure(nt) des capacités de préhension et de manipulation. Cette définition ouverte laisse la place à un grand nombre de combinaisons possibles, illustrées par la diversité des dispositifs expérimentaux et/ou commerciaux existants à ce jour [53].

Dans le cas le plus simple, lorsqu'on parle d'un *Manipulateur Mobile* (*Mobile Manipulator*), on parle souvent d'un système robotisé constitué d'un bras manipulateur à plusieurs degrés de liberté (*dll*) fixé sur une base mobile capable de se déplacer dans son environnement. Le succès de tels systèmes est lié à l'espace de travail quasiment infini qu'ils présentent. Quand un manipulateur industriel voit son espace de travail limité à quelques mètres cubes, le manipulateur mobile peut évoluer sur plusieurs centaines de mètres. Ils ouvrent donc un grand nombre de possibilités qui, pour la plupart, restent à explorer [53].

Ce premier chapitre a pour objectif de répertorier les informations fondamentales permettant de se familiariser avec les différents notions et concepts liés aux manipulateurs mobiles. Dans une première partie, quelques définitions et une présentation générale des manipulateurs mobiles seront données. Une deuxième partie aura pour but de décrire les caractéristiques techniques de ce type de robots. Dans un dernier volet, nous exposerons les domaines d'applications les plus connus où les manipulateurs mobiles ont pu prouver leur unicité.

L'utilisation des manipulateurs mobiles a offert la possibilité d'accomplir la plupart des tâches usuelles de la robotique qui requièrent, en même temps, des capacités de locomotion et de manipulation. Ces robots ont des applications dans plusieurs domaines tels que : *(i)* le domaine

[1] Un espace de travail est défini par la liste des positions et orientations accessibles et qui sont atteignables par le manipulateur à partir de la position actuelle de sa base. Les différents points qui appartiennent à l'espace de travail d'un manipulateur dépendent de la géométrie du manipulateur, des types de liaisons qui le composent et des limites articulaires de ces liaisons [26].

manufacturier pour accomplir des tâches de saisie et de transport d'objets, ou pour accomplir des tâches de manutention, *(ii)* dans le domaine militaire pour le déminage, etc.

Figure 1. Manipulateur mobile "MM-500".

Les possibilités d'application pour les manipulateurs mobiles sont diverses. Dans les dernières années, les environnements cibles et les applications potentielles de tels robots ont migrés des environnements industriels vers les environnements humains [45]. Parce qu'ils sont, particulièrement, mieux adaptés aux tâches humaines, ces robots sont présents actuellement dans les bureaux, les hôpitaux, les maisons pour l'assistance aux personnes handicapées et/ou personnes âgées, etc. [6].

1.2. Caractéristiques des manipulateurs mobiles

Les manipulateurs mobiles bénéficient de la complémentarité offerte par les deux sous-systèmes mécaniques qui les constituent. La base mobile augmente les possibilités de franchissement et d'évitement d'obstacles présents au niveau de l'espace de travail, ce qui permet d'accroitre, ainsi, les possibilités pour d'éventuelles manipulations. Sur le plan nature des tâches possibles, généralement limitées au simple transport d'objets pour une base mobile, un manipulateur permet d'étendre la diversité de ces tâches et d'offrir également des nouvelles potentialités (ouvrir une porte fermée, déplacer un objet, …).

Dans la littérature, plusieurs travaux sont portés sur les manipulateurs mobiles. Chaque manipulateur mobile présente des spécificités et des caractéristiques qui doivent être prisent en compte lorsque nous travaillons sur ce type de système. Les paragraphes suivants rassemblent les principales caractéristiques liées aux manipulateurs mobiles.

1.2.1. Autonomie

Pour un robot, l'autonomie peut être définie comme étant la capacité d'exécuter une tâche (opération ou plan d'opérations) sans intervention d'un opérateur et ce quelles que soient les

événements imprévus qui accompagnent son évolution. Il y a naturellement une limite à de telles variations au-delà de laquelle le robot s'avère impuissant à satisfaire son objectif final. Le degré d'autonomie se caractérise par la variété des situations que le robot peut appréhender, et se traduit par l'expression, plus ou moins, précise ou directive de la nature des tâches qui lui sont assignées. [29].

Un robot peut avoir plusieurs niveaux d'autonomie. La figure suivante montre ces différents niveaux d'autonomie.

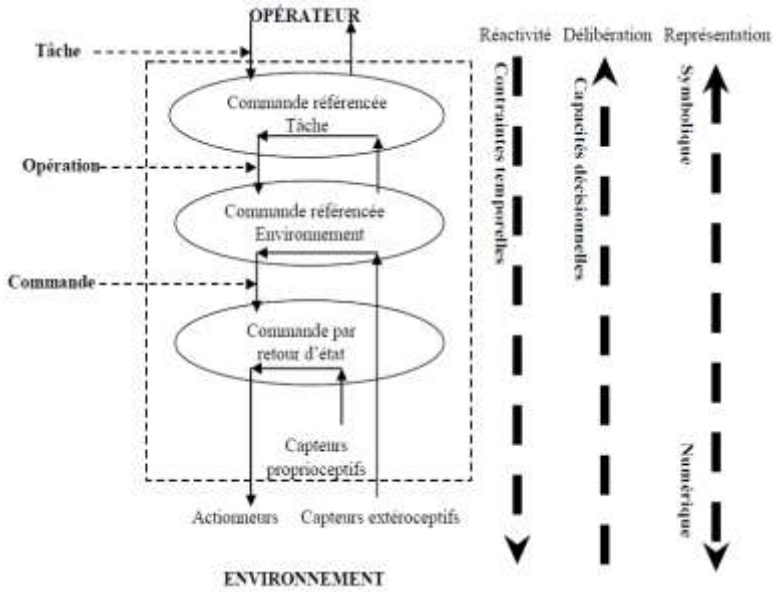


Figure 2. Différents niveaux d'autonomie d'un robot [21].

Le degré d'autonomie est parmi les enjeux essentiels de la robotique. Pour un robot, elle peut être vu comme une propriété agissant dans un environnement variable avec une intervention humaine réduite. À cet effet, deux axes symbolisent les besoins liés à l'autonomie [40] :

- *Capacité d'agir avec une intervention humaine réduite* : les objectifs fournis au robot sont généralement de très haut niveau avec un horizon temporel lointain. Par conséquent, des décisions nécessaires au bon déroulement d'une tâche et à la réalisation des objectifs visés doivent être prises par le robot. Les tâches à réaliser requièrent des capacités décisionnelles adaptées à leur haut niveau d'autonomie. Si les objectifs donnés ne peuvent pas être exécutés directement, le robot doit être capable de déduire les opérations à effectuer ainsi que leur ordre pour atteindre ses objectifs fixés. Pour assurer de tels contraintes, les robots embarquent des outils complexes prenant en compte le temps, les ressources ou encore l'incertitude du monde.
- *Variabilité de l'environnement* : le plan d'opérations du robot est menacé, en permanence, par les changements présents dans l'environnement. Il doit donc exister des mécanismes offrant une certaine capacité de corriger un service dans une situation adverse non prévisible issue de l'environnement (obstacle inattendu, porte fermée, etc.). Même s'il rencontre de nouvelles

situations inattendues, un robot autonome doit être capable d'accomplir correctement sa tâche sans intervention humaine.

1.2.2. Non-holonomie

Pour un manipulateur mobile, la non-holonomie est une propriété liée à la partie base mobile. Un système mobile holonome peut se mouvoir dans toutes les directions depuis sa position courante, tandis qu'un système non-holonome est soumis à des contraintes sur les vitesses des différents corps qui le composent. De ce fait, une base mobile non-holonome ne peut pas se déplacer instantanément que dans certaines directions (Figure 3). Par exemple, une voiture ne peut se garer sans effectuer de manœuvres. Cette propriété illustre la nécessité de faire des manœuvres en voiture afin de pouvoir se garer.

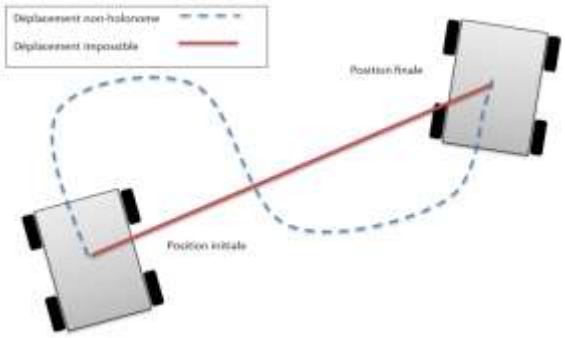


Figure 3. Déplacement d'une base mobile non-holonome.

1.2.3. Capteurs utilisés

Afin d'accomplir ses tâches, un manipulateur mobile doit connaître son environnement. Cette connaissance est fournie par des capteurs qui sont installés sur le robot ou dans son environnement. Les types de capteurs utilisés et leurs positions représentent un critère important dans le choix d'un manipulateur mobile pour la réalisation d'une tâche particulière.

Traditionnellement, on classe les capteurs en deux catégories selon qu'ils mesurent l'état interne du robot lui-même ou l'état de son environnement. Ces capteurs lui permettent de se localiser dans son environnement, d'appréhender ce qui l'entoure, de naviguer en évitant des obstacles se trouvant sur son chemin et, d'agir sur l'environnement avec la précision et l'efficacité voulues. Les capteurs lui permettent aussi de prendre des décisions et d'avoir une certaine autonomie, pour s'acquitter de la meilleure façon possible des tâches, qui demandent, souvent, beaucoup de précision lors de leur exécution. De ce fait, le robot est équipé, en général, de deux types de capteurs :

- *Capteurs proprioceptifs* : ces capteurs mesurent l'état mécanique interne du robot, tels que les capteurs odométriques, les capteurs de positions des axes du manipulateur, les capteurs de vitesse ou d'accélération, les capteurs de charge des batteries, etc.
- *Capteurs extéroceptifs :* ces capteurs renseignant sur l'état de l'environnement, donc de ce qui est extérieur au robot lui-même. On peut citer par exemple les capteurs de détection de présence, les systèmes de mesure de proximité et de distance (par des procédés acoustiques, optiques ou radioélectriques), les capteurs d'effort, les capteurs de vision, les systèmes de positionnement et de localisation, etc.

1.2.4. Types d'articulations pour le manipulateur

Pour la partie mécanique d'un manipulateur, nous distinguons deux types d'articulations (liaisons) [3] :

- *Articulation rotoïde :* c'est une articulation de type pivot, ayant comme principe la réduction du mouvement entre deux corps à une rotation autour d'un axe qui leurs est commun, ce qui donne comme résultante un angle de rotation autour de cet axe.
- *Articulation prismatique :* c'est une articulation de type glissière, réduisant le mouvement entre corps à une translation le long d'un axe commun, ce qui signifie qu'il se produira un déplacement linéaire mesuré par une distance le long de cet axe.

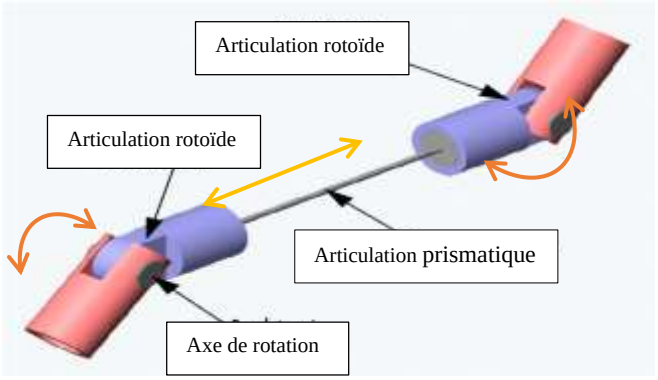


Figure 4. Types d'articulations pour un manipulateur.

1.2.5. Degrés de liberté

Pour un système robotisé, le degré de liberté est défini par le nombre de paramètres indépendants qui fixent la situation de l'effecteur (position et orientation). Il représente le nombre de mouvements élémentaires (rotation et translation) par rapport à un repère de base.

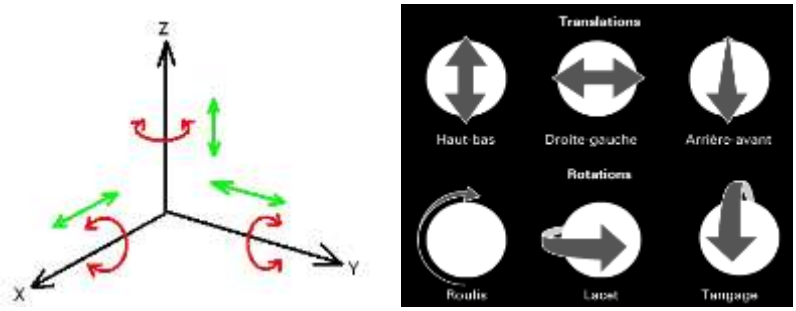


Figure 5. Degrés de liberté.

1.3. Domaines d'applications des manipulateurs mobiles

Les manipulateurs mobiles sont conçus pour accomplir des tâches de nature différente, laborieuses pour l'être humain, et qui nécessitent, parfois, une précision accrue. Ils sont également capables d'opérer dans des environnements qui présentent un danger pour la présence humaine (radiations, gaz toxiques, etc.). Nous citons dans ce qui suit quelques exemples de l'utilisation des manipulateurs mobiles dans des domaines différents en mettant l'accent sur des travaux de recherche récents.

1.3.1. Domaine spatial

Les manipulateurs mobiles ont leur place dans l'exploration de l'espace. Comme exemple, on peut citer le robot mobile japonais Micro5 (Figure 6), auquel on a intégré un bras manipulateur. Ce robot a été conçu pour pouvoir évoluer sur sols accidentés. Il est, cependant, utilisé dans des opérations de collecte d'échantillons ou d'insertion d'équipements scientifiques dans le sol [36].

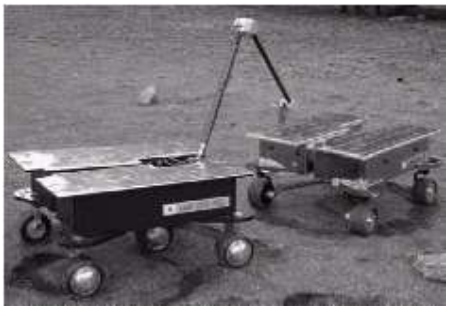


Figure 6. Robot Micro5.

1.3.2. Domaine agricole

Dans le but de faciliter des travaux lassants et épuisants pour l'homme dans le domaine agricole, le manipulateur mobile *AGROBOT* du *Laboratory For Integrated Advanced Robotics* de l'université de Gênes en Italie (Figure 7) a été conçu pour effectuer des travaux agricoles sous serres pour des tâches de pulvérisation de produits, et d'arrachage des fruits abîmés grâce à un système de vision stéréoscopique en couleur [23].

Figure 7. Robot Agrobot.

1.3.3. Domaine médical

Dans le but de remplir des tâches en milieu hospitalier où à domicile, les membres du *Intelligent Robotics Laboratory* de l'université de Vanderbilt ont conçu le robot *Helpmate* (Figure 8). L'utilisation d'une caméra montée sur le manipulateur de ce robot, ainsi que des capteurs vocaux et tactiles, est sensée offrir certaine autonomie aux personnes handicapées [49].

Figure 8. Robot Helpmate.

1.3.4. Domaine manufacturier

Les chercheurs dans l'*Institute for Real-time Computer Systems And Robotics* de l'université de *Karlsruhe* en Allemagne ont conçu le robot Kamro (Figure 9). Ce robot est destiné à accomplir des tâches de façon autonome en environnement industriel. La plateforme mobile omnidirectionnelle de ce robot lui permet une grande facilité de mouvement. Elle est dotée également de plusieurs capteurs à ultrason. Ce manipulateur mobile est constitué de deux bras équipés de capteurs d'effort six axes, ainsi que deux caméras se trouvant sur chacun des effecteurs [37].

Figure 9. Robot Kamro.

1.4. Conclusion

L’objectif de ce chapitre était de se familiariser avec les concepts, les aspects structurels et mécaniques liés aux manipulateurs mobiles en générale. Nous avons, à cet effet, essayé de donner un aperçu général sur le domaine des manipulateurs mobiles. En outre, nous avons présenté les caractéristiques principales de ce type de robots ainsi qu’une liste non exhaustive des domaines d'utilisation de ces robots avec comme exemples des travaux de recherche récents.

Dans le prochain chapitre de ce mémoire, nous allons présenter un état de l’art sur les architectures de contrôle pour ce type de robots ainsi qu’une classification des approches de contrôle proposées dans la littérature.

CHAPITRE 2
ARCHITECTURES ET APPROCHES DE CONTRÔLE DES MANIPULATEURS MOBILES

2.1. Introduction

Dans un système robotisé, nous distinguons deux éléments essentiels (*i*) le *robot* qui est constitué d'une structure mécanique poly-articulaire et des actionneurs et (*ii*) le *contrôleur* qui décrit une architecture matérielle et logicielle apportant l'intelligence nécessaire au robot pour qu'il puisse effectuer les tâches pour lesquelles il a été conçu. La partie matérielle du contrôleur comprend les cartes électroniques permettant au logiciel de s'exécuter et des cartes d'entrée/sortie qui font le lien avec l'environnement extérieur. La partie logicielle est chargée d'effectuer le traitement des informations provenant des capteurs proprioceptifs et extéroceptifs, de gérer la communication et la synchronisation des processus et de prendre les décisions nécessaires au bon déroulement de la tâche en cours [39]. Dans le cadre de ce mémoire, nous nous sommes intéressés uniquement aux aspects logiciels des contrôleurs de robots.

Ces dernières années, plusieurs travaux dans la littérature étaient intéressés au contrôle des manipulateurs mobile autonomes. Ces travaux ont mené au développement de plusieurs approches de contrôle dédiées à ce type de robots [27].

Une architecture de contrôle est considérée comme une structure logicielle constituée d'un ou plusieurs niveaux de traitement. Chaque niveau de traitement est responsable de la prise en charge d'un ou plusieurs aspects assurant le processus de contrôle d'un système robotisé. Une approche de contrôle représente une panoplie de modèles mathématiques et d'outils informatiques, et la manière dont on les exploite pour garantir le comportement souhaité.

Ce chapitre commence par la définition d'un contrôleur. Quelques définitions et une classification des architectures de contrôle seront présentées. Ensuite, quelques travaux de recherche, qui ont porté sur le développement des approches de contrôle dédiées aux manipulateurs mobiles, sont invoqués selon le type d'approche et les techniques sollicités, tout en mettant l'accent sur les approches basées sur les systèmes multi-agents.

2.2. Contrôleur de robot

Un *contrôleur de robot* est la partie qui offre l'intelligence dans un système robotisé [16]. Un contrôleur décrit une architecture matérielle et logicielle qui traite les informations et gère les données, issues des capteurs, sur l'environnement pour contrôler le robot. Ce contrôle correspond tant à la commande qu'à la supervision du robot [39]. [41] définit le contrôle d'un robot par le processus de récolte d'informations sur l'environnement à travers des capteurs, de traitement de ces informations dans le but de prise de décisions sur les actions et, ensuite, l'exécution de ces opérations.

Dans la littérature, plusieurs travaux se sont intéressés à la proposition de modèles de contrôleurs pour les manipulateurs mobiles. Chaque proposition donne une manière spécifique à la résolution du problème de contrôle. Certaines méthodes proposent un découplage total entre les deux sous-systèmes (manipulateur et base mobile). Lors de la réalisation d'une tâche qui nécessite l'intervention des deux sous-systèmes, le contrôleur enchaine le contrôle d'une façon séquentielle entre les deux entités disjointes. Il existe également des modèles pour synchroniser le contrôle de la base mobile et du bras manipulateur.

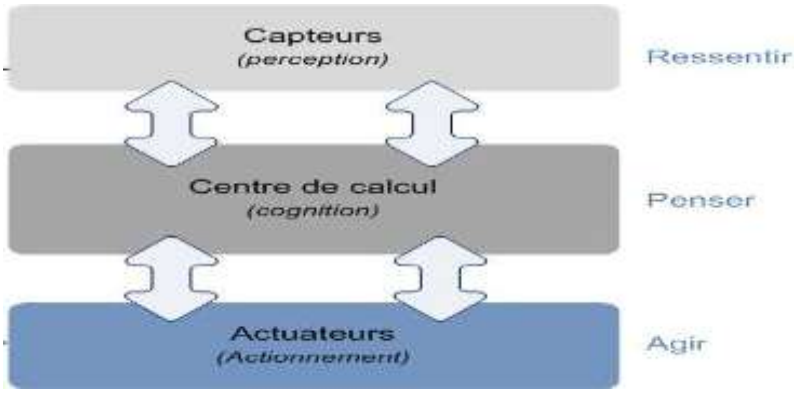


Figure 10. Schéma synoptique pour le processus de contrôle.

2.3. Architectures de contrôle

Plusieurs définitions pour une architecture de contrôle ont été évoquées dans la littérature. Dean et Wellman [12] définissent une architecture de contrôle comme étant un ensemble de composantes architecturales et leurs manières d'interagir. Albus [5] définit une architecture comme une description de la façon dont un système est construit à partir de composantes de base et la façon dont ces éléments s'assemblent et interagissent pour former un ensemble. Une architecture de contrôle a comme objectif principal la simplification du développement des différents modules élémentaires et leur composition, nécessaires pour assurer des comportements complexes [10]. Enfin, Mataric [41] [42] définit une architecture robotique de contrôle comme étant une méthodologie qui fournit un moyen d'organiser un système de contrôle. Toutefois, en plus de fournir la structure, il impose des contraintes sur la façon dont le problème de contrôle peut être résolu.

De ce qui précède, une définition globale pour une architecture de contrôle peut être alors donnée comme suit : *Un ensemble structuré de fonctions élémentaires que le robot peut réaliser. Le contrôle est l'entité qui manipule les différents aspects de cette architecture*.

Dans la finalité de réalisation des objectifs assignés par un opérateur, chaque concepteur propose sa propre démarche pour la structuration des fonctions internes du robot, et l'organisation des opérations à exécuter et des tâches à accomplir par ce dernier.

L'organisation d'une architecture de contrôle de robots autonomes est une tâche très importante pour le succès et la réussite d'une intervention du robot, car le succès d'une telle intervention repose et dépend du comportement du robot [17]. Le choix de cette architecture dépondra des outils et des plateformes (hard et soft) utilisés :

- *Hardware* : représente les spécificités techniques du robot en termes d'actionneurs, de capteurs et aussi de calculateurs.
- *Software* : comporte les plateformes de développement et d'exécution utilisées.

Plusieurs chercheurs se sont consacrés au développement des architectures de contrôle de robots autonomes. Les différentes architectures robotiques définies dans la littérature ne diffèrent pas forcement par les méthodes élémentaires employées, mais plutôt, par leurs agencements et leurs relations. Il y a lieu de préciser qu'aucune architecture n'est parfaite pour répondre à toutes les tâches, et que différentes tâches ont différents critères de succès [46]. Dans ce qui suit, une classification, en trois grandes catégories, selon le mode de fonctionnement est présentée.

2.3.1. Architectures délibératives

Ces architectures, connues aussi sous les *architectures classiques* ou *hiérarchiques*, sont les premières à être utilisées en robotique. Elles visent à reproduire le mode de raisonnement humain, en adaptant le paradigme *Sense-Model-Plan-Act*. L'architecture est, alors, constituée d'un ensemble de modules qui communiquent et échangent des informations. Le contrôle s'obtient par la synchronisation de ces divers modules [47]. Le traitement est décomposé en une série de *composantes fonctionnelles successives* (ou *modules*) décrites par le diagramme de la figure suivante.

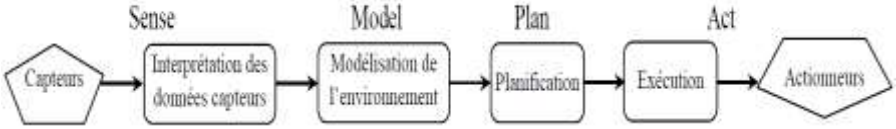


Figure 11. Composantes fonctionnelles d'une architecture délibérative.

L'avantage de cette architecture réside dans la possibilité d'intégrer des raisonnements qui s'appuient sur des modèles complets de l'environnement (cartes, etc.). Elle permet également de trouver une séquence optimale d'opérations garantissant l'accomplissement d'une tâche, avant même que le robot commence son exécution. Son principal inconvénient provient de la faible vitesse de réaction aux changements légers des situations. ces architectures sont, en effet, incapables de répondre à des exigences en temps-réel. Un autre inconvénient majeur est que ces systèmes sont, en général, peu robustes de par leur modèle centralisé. Comme les informations sont traitées de manière séquentielle, une défaillance dans l'une de ces composantes peut provoquer le blocage complet de l'architecture [29].

2.3.2. Architectures réactives (comportementales)

Un comportement est un module simple qui reçoit des entrées provenant des capteurs du robot et/ou d'autres comportements et fournit, en sortie, des commandes aux actionneurs et/ou d'autres comportements dans le système [17]. Ce type d'architectures est une collection de *comportements*. Ces derniers s'exécutent en concurrence, en agissant en parallèle, pour atteindre des objectifs indépendants [31]. Dans cette architecture, le contrôle est réalisé avec les interactions entre les différents comportements qui sont indépendant les uns des autres. Plusieurs comportements doivent être exécutés simultanément et de manière asynchrone, alors qu'un mécanisme d'arbitrage (Coordinateur) doit être défini lorsque plus d'un comportement se charge de déterminer les actions à prendre [47]. La figure suivante schématise le principe de ce type d'architectures.

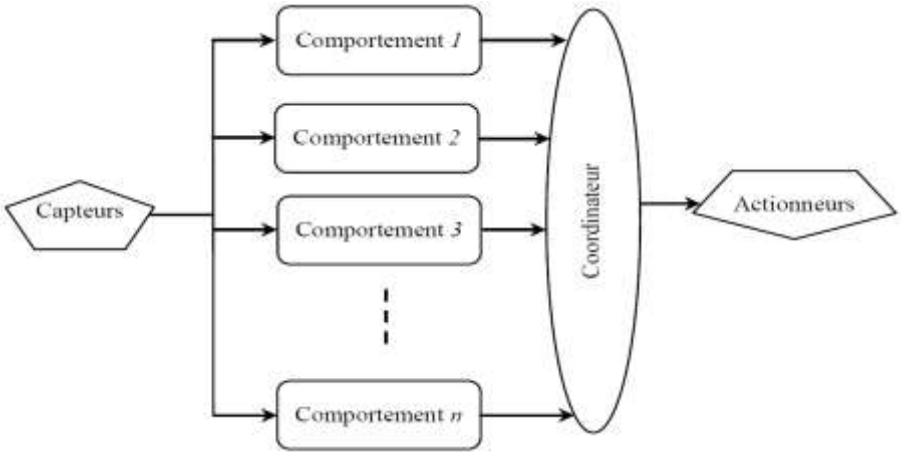


Figure 12. Principe des architectures de contrôle comportementales.

Cette architecture est efficace lorsque l'environnement du robot, à cause des changements fréquents, ne peut pas être modélisé avec précision. Comme il n'existe pas un modèle de l'environnement et du fait que les comportements soient simples, la puissance de calcul et la taille de mémoire nécessaires sont considérablement réduites. Enfin, la caractéristique la plus puissante est que l'intelligence du système peut être améliorée progressivement par la création de nouveaux comportements [29].

De l'autre côté, il est très difficile de concevoir des robots purement réactifs qui peuvent apprendre de l'expérience et améliorer leurs performances au cours du temps. De plus, ne pas considérer les répercussions des actions, limite la complexité des buts qui peuvent être pris en compte. Il est nécessaire alors de concevoir un système supérieur qui gère au mieux cette réactivité pour pouvoir implémenter un comportement générique et sophistiqué [47].

2.3.3. Architectures hybrides

De ce qui précède, nous avons pu constater que les architectures réactives ont prouvé leur efficacité dans des environnements dynamiques et complexes. Néanmoins, ce type d'architectures ne prend pas en considération la délibération ni la modélisation de l'environnement lorsque la tâche à exécuter l'exige. De plus, nous avons noté que les architectures classiques ou délibératives ne représentent pas un bon choix pour la réalisation des tâches dans des environnements complexes. Ainsi, la création d'architectures hybrides est une réponse aux limitations constatées de chacun des types d'architectures de contrôle développés par le passé. L'idée générale est de combiner les différentes approches dans une seule et même architecture appelée *Architecture hybride* (Figure. 13) [29].

La fusion des deux architectures, réactive et délibérative, permet de compenser les inconvénients et les limites de chacune, tout en préservant leurs avantages. Les architectures

hybrides incluent une partie délibérative pour la modélisation de l'environnement, le raisonnement et la génération de plans, et une deuxième partie réactive, responsable de l'exécution des plans générés et de la réaction aux situations imprévisibles pouvant surgir [29].

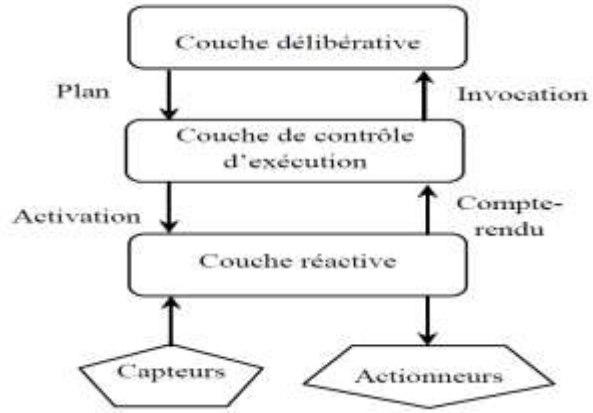


Figure 13. Diagramme d'une architecture de contrôle hybride.

2.4. Approches de contrôle des manipulateurs mobiles

Une approche de contrôle est considérée comme étant un ensemble de modèles et techniques utilisés pour réussir le processus de contrôle pour un manipulateur mobile. Nous définissons une approche de contrôle comme la méthodologie adoptée lors du traitement des données en entrée, issues des différents capteurs du robot, pour offrir, en sortie, les consignes de contrôle.

Dans ce qui suit, nous allons présenter une classification en deux types d'approches de contrôle selon les techniques utilisées pour assurer le contrôle des manipulateurs mobiles. Un premier type d'approches qui utilisent les modèles mathématiques pour contrôler le robot. Le deuxième type d'approches sont basées sur les techniques de l'intelligence artificielle. Colle et ses collègues [57] ont montré que l'utilisation de l'IAD *(Intelligence Artificielle Distribué)* pour la résolution de problèmes, offre une alternative à l'utilisation des modèles mathématiques pour commander un manipulateur mobile.

Dans [27], nous avons essayé de synthétiser les différentes approches de contrôle de robots manipulateurs mobiles autonomes. Nous avons classifié ces approches, en deux grandes classes données comme suit ;

- L'approche roboticienne classique.
- L'approche basée sur les systèmes multi-agents.

2.4.1. Approche roboticienne

Cette première classe, considérée comme classique, est basée sur l'étude des modèles mathématiques des deux sous-systèmes mécaniques qui constituent un manipulateur mobile (bras manipulateur et base mobile) [7] [48]. Dans un *système robotisé,* la modélisation mathématique a

pour but de représenter au mieux le robot dans son environnement dont la finalité est d'offrir un contrôle plus précis et plus efficace lors de l'accomplissement de tâches.

Contrôler un manipulateur mobile revient à calculer les mouvements articulaires du bras manipulateur ainsi que les déplacements de la base mobile. Pour la réalisation de ces mouvements, il faut étudier le *Modèle Géométrique Direct* (*MGD*) et le *Modèle Géométrique Inverse* (*MGI*) du robot.

Définir la situation d'un objet libre représenté dans un espace à trois dimensions nécessite, dans le cas général, la connaissance de six paramètres indépendants. Trois de ces paramètres définissent la position d'un point de l'objet, et les trois autres grandeurs déterminent son orientation autour du point précédent [24]. Ces coordonnées, en nombre minimal, seront englobées sous le vocable "situation". Les six coordonnées indépendantes représentant la situation de l'*effecteur*, forment un vecteur *X* dit vecteur des coordonnées opérationnelles. Les trois premières valeurs (x_E, y_E, z_E) représentent les coordonnées cartésiennes dans l'espace tridimensionnel pour l'*effecteur*. Les trois autres $(\psi_E, \theta_E, \varphi_E)$ sont utilisées pour exprimer l'orientation suivant les Angles d'Euler.

2.4.1.1. Modèle Géométrique Direct (MGD)

Le *Modèle Géométrique Direct (MGD)* permet de calculer la configuration de l'effecteur du manipulateur mobile en fonction de la position de chaque articulation motorisée (liaison) du manipulateur ainsi que de la position et orientation de la base mobile. Il s'écrit alors :

$$X=MGD(q, Base)$$

tels que :

- *X* : représente la situation de l'effecteur donnée par $(x_E, y_E, z_E, \psi_E, \theta_E, \varphi_E)$.
- *q* : le vecteur des positions articulaires $(q_1, q_2, \ldots, q_n)$ d'un manipulateur à *n* articulations.
- *Base* : les coordonnées ainsi que l'orientation de la base mobile données par (x_B, y_B, θ_B).

La situation de l'effecteur en fonction des différentes coordonnées articulaires est calculée selon la notation MDH (*Denavit-Hartenberg Modifiée*) [35].

La sortie fournie par le MGD est le vecteur *X* défini par $(x_E, y_E, z_E, \psi_E, \theta_E, \varphi_E)^T$. Les trois premières valeurs sont les coordonnées de l'effecteur et les trois dernières représentent son orientation. Ce vecteur est défini dans un référentiel fixe, le repère absolu (R_A), où la situation du robot est représentée dans l'espace à trois dimensions. Le repère absolu ainsi que les différents repères sollicités sont présentés dans la figure 14.

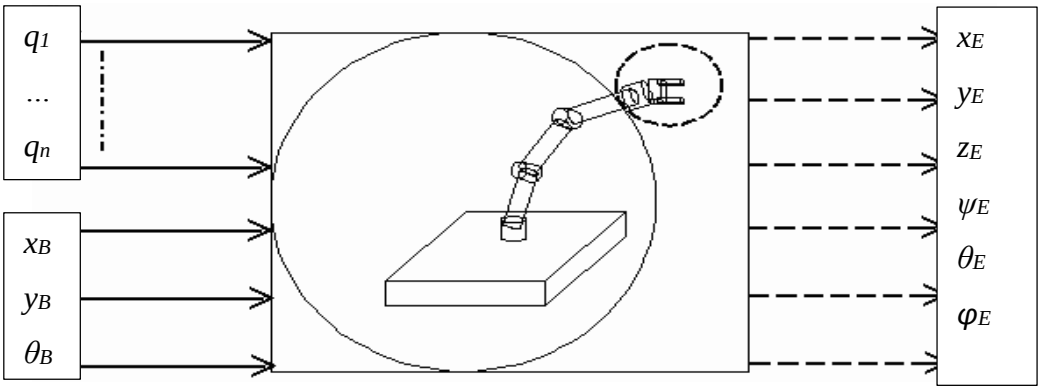


Figure 14. Calcul du Modèle Géométrique Direct (MGD) d'un manipulateur mobile.

La méthode utilisée, lors du calcul du *MGD*, fait appel à différentes matrices relatives aux différents repères (Figure 15). Cela signifie que chaque repère se déplace relativement à un autre. De ce fait, *le Repère lié à la Base mobile (R_B)* est considéré en mouvement par rapport au *Repère Absolu (R_A)*, *le Repère lié au Manipulateur (R_M)* est considéré en mouvement par rapport à R_B et le *Repère lié à l'Effecteur* (R_E) se déplace relativement à R_M. Cette approche est particulièrement structurée, elle est très implicite en termes de représentation et de vision géométrique dans l'espace, dans le sens où la notion de relativité apparaît grâce à la disposition des matrices [29].

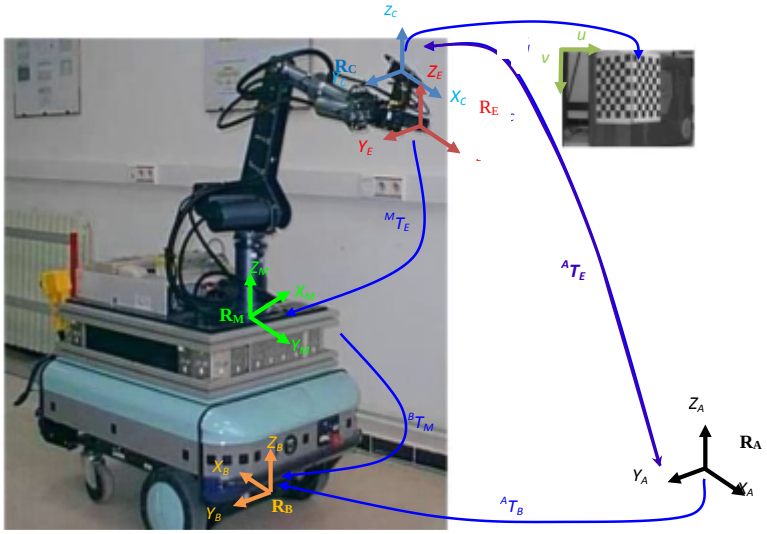


Figure 15. Principaux repères liés à un manipulateur mobile.

La situation de l'effecteur est donnée, dans R_A, par la relation suivante :

$$^{A}T_E = {}^{A}T_B * {}^{B}T_M * {}^{M}T_E$$

telles que :

- ${}^{A}T_E$: la matrice de transformation qui définit R_E dans R_A.
- ${}^{A}T_B$: la matrice de transformation définissant R_B dans R_A.
- ${}^{B}T_M$: la matrice de transformation qui définit R_M dans R_B.

- ${}^{M}T_{E}$: la matrice de transformation définissant R_E dans R_M.

Grâce à la matrice de transformation ${}^{A}T_{E}$, la situation de l'effecteur peut être déduite dans le repère absolu R_A. La figure suivante (*Figure 15*) montre d'une façon implicite l'ajustement des différents repères par rapport au manipulateur mobile.

2.4.1.2. Modèle Géométrique Inverse (MGI)

Le *MGD* admet une fonction inverse qui consiste à déterminer la configuration des liaisons articulaires ainsi que celle de la base mobile en fonction de la situation de l'effecteur, c'est le *Modèle Géométrique Inverse* (*MGI*). Ce modèle permet d'exprimer la configuration du système robotique en fonction de la situation de l'*effecteur* [3]. Pour le manipulateur mobile, le *MGI* s'écrit alors sous la forme :

$$(Base, q)=MGI(X)$$

Comme la définition de ce type de modèle le stipule, le *MGI* d'un manipulateur mobile quelconque évoluant dans un espace tridimensionnel, a comme but de calculer ses coordonnées généralisées $(x_B, y_B, \theta_B, q_1, q_2, ..., q_n)^T$, en fonction de la situation imposée à son *effecteur* $(x_E, y_E, z_E, \psi_E, \theta_E, \varphi_E)^T$ dans le repère R_A. Il convient, cependant, de noter que le *MGI* présente une infinité de solutions. Cela est dû à la *redondance géométrique.*

La redondance géométrique exprime le fait que le nombre de coordonnées généralisées ν est strictement supérieur au nombre de ddl μ de l'effecteur. L'ordre de redondance géométrique est égal à ν−μ. Elle signifie que pour une situation donnée de l'effecteur, il existe une infinité de configurations du système. Il est donc considéré comme géométriquement redondant si μ<ν. L'ordre de cette redondance géométrique étant ν−μ. Par contre, si μ=ν, le système mécanique n'est pas géométriquement redondant, alors que la condition μ>ν s'avère impossible. La figure suivante (*Figure 16*) représente un schéma de calcul du *MGI*.

Contrairement au *MGD* d'un manipulateur mobile, Il n'y a pas de lois particulières pour calculer son inverse. Cependant, c'est aux concepteurs du contrôleur de choisir les stratégies propres au type de systèmes à étudier (le bras manipulateur et la base mobile) [3].

Le *MGI* du RobuTER/ULM, qui évoluant dans un espace tridimensionnel, a comme but de calculer la configuration du robot $(x_B, y_B, \theta_B, q_1, q_2, q_3, q_4, q_5, q_6)$, en fonction de la situation imposée à son effecteur $(x_E, y_E, z_E, \psi_E, \theta_E, \varphi_E)$ dans le repère R_A. Tandis que le bras manipulateur *ULM* est composé de 6 ddl $(q_1, ..., q_6)$ et que le nombre de coordonnées généralisées s'accroît en ajoutant celles de la base mobile (x_B, y_B, θ_B), le manipulateur mobile *RobuTER/ULM* est considéré comme géométriquement redondant vis-à-vis les tâches à accomplir. Afin de faire mouvoir

RobuTER/ULM à une situation donnée, il faut calculer les différentes q_i (i=1...6) qui correspondent à cette situation en utilisant le *MGI* (*Figure 16*).

Figure 16. Calcul du Modèle Géométrique Inverse (MGI)

Contrairement au *MGD*, son inverse n'a pas de règles de calcul spécifiques. Par conséquent, une stratégie propre au *RobuTER/ULM* a été employée dans les travaux de [29]. La solution qui a été proposée découple partiellement le système robotique. Elle commence par le mouvement de la base mobile, à une nouvelle situation $Base_{Fin}$, en prenant en compte les contraintes de non-holonomie ainsi que celles liées à l'environnement (évitement d'obstacles), de sorte à ce qu'elle puisse placer le manipulateur dans une situation lui permettant d'atteindre la situation opérationnelle imposée ($Effector_{Fin}$). Puis, le modèle géométrique du manipulateur est inversé en utilisant une méthode de calcul analytique décrite dans [24] et donnée par (*Figure 17)*, pour, enfin, placer l'effecteur dans la situation désirée.

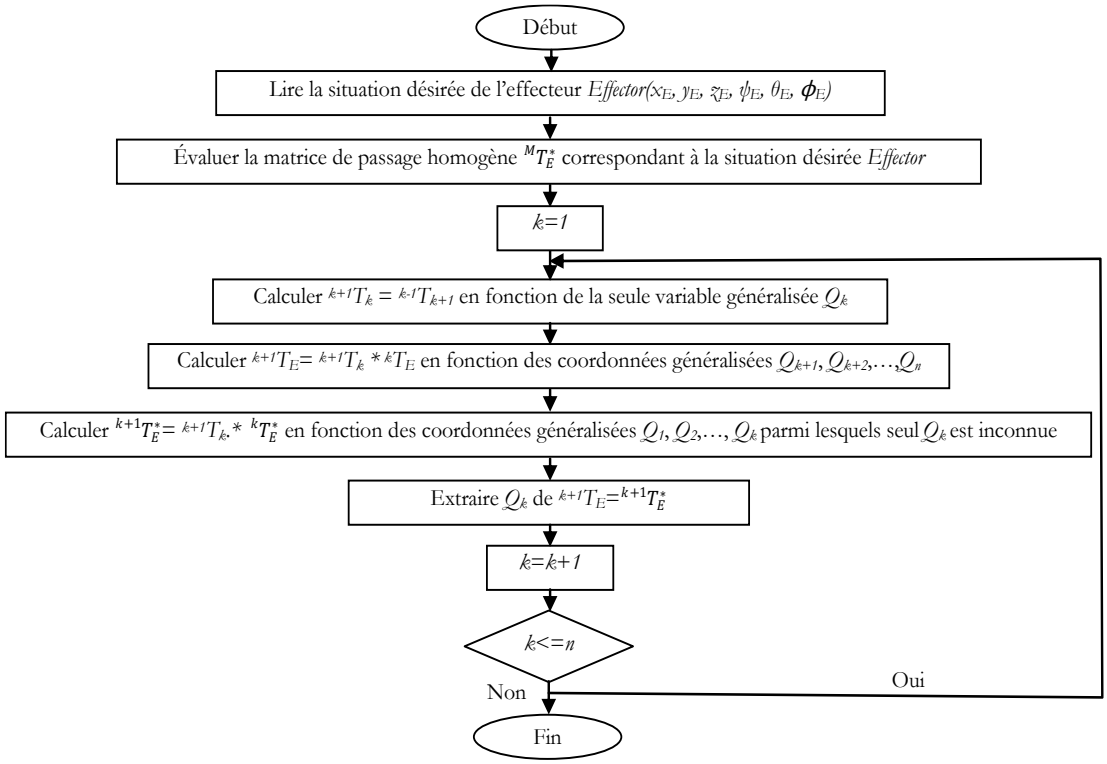


Figure 17. Méthode d'inversion du modèle géométrique utilisé par [24].

La situation finale de l'effecteur $Effector_{Fin}(x_{EFin}, y_{EFin}, z_{EFin}, \psi_{EFin}, \theta_{EFin}, \varphi_{EFin})$ est complètement définie dans le repère R_A. Mais, la situation de la base mobile $Base_{Fin}$ $(x_{BFin}, y_{BFin}, \theta_{BFin})$, ainsi que la configuration du manipulateur $(q_1, q_2, q_3, q_4, q_5, q_6)$ peuvent varier de manière significative (*Figure 18*). De plus, les contraintes de non-holonomie de la base mobile différentielle peuvent avoir un effet significatif sur la forme de la trajectoire requise pour atteindre $Effector_{Fin}$.

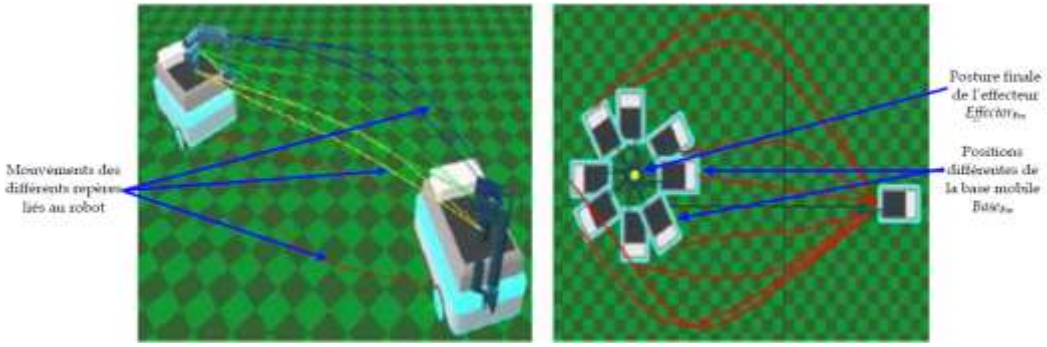


Figure 18. Variations possibles de $Base_{Fin}$ et leurs influences sur la configuration du manipulateur pour atteindre $Effector_{Fin}$ [2]

2.4.1.3. Discussion de l'approche roboticienne

Le long de notre étude bibliographique, nous avons constaté que la mise en œuvre d'une approche roboticienne, nécessite le recours aux modèles mathématiques du système étudié. Donc une certaine expertise sur la mécanique du robot est requise, en ce qui concerne (*i*) les paramètres *MDH*, (*ii*) le *MGD*, (*iii*) le *MGI*.

L'utilisation des modèles mathématiques donne des résultats exacts et offre un contrôle assez précis. Il faut, cependant, noter que les méthodes de calcul des modèles directs représentaient des lois génériques, alors que les modèles inverses ont été construits selon la composition du manipulateur mobile.

Le mouvement de la base mobile ajoute au robot des nouveaux ddl, que le concepteur du contrôleur doit prendre en considération lors de la modélisation. La contrainte de non-holonomie d'une base mobile ne peut être résolue par le modèle géométrique nécessitant, ainsi, d'aller vers l'étude du modèle cinématique et de considérer l'aspect vitesse.

Utiliser les modèles mathématiques, pour contrôler un manipulateur produit de bons résultats pour des tâches répétitives dans des environnements connus, ce qui ne correspond pas aux conditions d'utilisation dans certaines domaines de la robotique ou l'environnement n'est pas complètement connu mais, en revanche, évolutif. Ces modèles sont, dans la majorité des cas, calculés hors ligne et sont incapables de s'adapter à des modifications dans la structure mécanique du robot, par exemple un disfonctionnement d'un des axes du robot, sans l'addition de modes spécifiques de traitement de panne [13] [57].

Néanmoins, il faut préciser que l'utilisation des modèles mathématiques pour contrôler un manipulateur est très répondue en robotique industrielle. Cela est justifié par le fait que le système robotique évolue dans un environnement contrôlé et aussi puisque il est utilisé, souvent, pour des tâches répétitives. Dans ce dernier cas, lorsque le robot est amené à répéter une trajectoire des milliers de fois, le calcul hors ligne des modèles est adopté avec la possibilité d'optimiser le temps écoulé et/ou l'énergie dépensée lors de la réalisation d'une tâche.

2.4.2. Approche multi-agents

Pour contrôler un manipulateur mobile, l'approche générale, présentée précédemment, consiste à calculer les modèles géométriques et cinématiques. Cette approche produit de bons résultats pour des tâches répétitives dans des environnements contrôlés. Cependant, elle présente l'inconvénient de temps de calcul assez important lorsqu'il s'agit d'un environnement évolutif et qui n'est pas complètement connu. Ce qui rend cette approche inefficace dans un environnement qui éprouve des changements fréquents et lorsque le nombre de ddl du robot est considérable.

En général, les modèles mathématiques sont calculés hors ligne et sont incapables de s'adapter à des changements dans la structure mécanique de la machine, tels que le disfonctionnement d'un ou plusieurs articulations par exemple, sans l'addition de modes spécifiques de traitement de pannes. Lorsqu'une panne est survenue, la possibilité d'offrir un *service minimum*, en attendant la réparation, est un élément important de la qualité de service [57].

Lorsqu'il s'agit de problèmes complexes avec de dimensions importantes, la résolution des problèmes de contrôle pour un manipulateur mobile est très difficile en utilisant les modèles mathématiques traditionnels. Les techniques d'*intelligence artificielle distribuée (IAD)* proposent des solutions simples à entreprendre. Il s'agit de résoudre des sous-problèmes localement en utilisant des agents autonomes ce qui évite l'implémentation d'une gestion centralisée [13].

Dans la littérature, les techniques de l'intelligence artificielle ont été exploitées, dans plusieurs travaux, pour proposer des méthodes de résolution de problèmes complexes qui permettent de s'affranchir de la connaissance des modèles mathématiques du robot [25] [15].

En informatique, les techniques de calcul, qui sont basés sur des modèles centralisés et qui offrent un traitement séquentiel de l'information, évoluent vers des approches distribuées, qui proposent un traitement parallèle. Cette tendance a conduit vers les agents autonomes, capables de réaliser des tâches individuelles sans aucune intervention externe. Les *SMA* sont utilisés pour résoudre des problèmes complexes qui ne sont pas solvables, ou difficilement abordables, si on considère une seule unité de résolution avec sa vision limitée du problème dans sa globalité [18].

Dans ce qui suit, nous commençons par donner la définition d'un agent, ses propriétés ainsi que ses différents types existant. Ensuite, les systèmes multi-agents (*SMA*) sont abordés avec un

résumé sur leurs domaines d'applications et les avantages offerts par leur utilisation. Enfin, nous présentons une étude bibliographique sur les travaux qui ont eu recours aux *SMA* pour le contrôle des systèmes robotisés.

2.4.2.1. Définition d'un agent

Le concept des agents représente le paradigme le plus important pour le développement de software depuis l'orienté objet [43]. Le mot *agent* est utilisé dans plusieurs domaines, ce qui fait que plusieurs définitions lui sont attachées. En informatique, le concept d'agent est défini de manières différentes. La définition que nous avons adoptée, et qui semble couvrir les caractéristiques des agents que nous avons exploités, est celle proposée par [34 et [54]. Un agent est une entité logicielle. Il est considéré comme un système informatique, *situé* dans un environnement, qui agit d'une façon *autonome* et *flexible* pour atteindre les objectifs pour lesquels il a été conçu. Les concepts clés dans cette définition sont *(i)* situé dans un environnement, *(ii)* l'autonomie d'action et *(iii)* la flexibilité. [11] ont donnés les définitions suivantes :

- *Situé :* il signifie que l'agent peut recevoir des entrées sensorielles provenant de son environnement et qu'il peut effectuer des actions qui sont susceptibles de changer cet environnement.
- *Autonome* : elle représente la capacité d'agir sans l'intervention d'un tiers (humain ou agent) et le contrôle de son état interne ainsi que ses propres actions.
- *Flexible* : pour un agent, c'est le fait qu'il soit :
    - *capable de répondre à temps* : élaborer une réponse, à des stimuli issus de son environnement, dans les temps requis.
    - *proactif* : tout en étant capable de prendre l'initiative, un agent devra exhiber un comportement opportuniste, dirigé par ses buts ou sa fonction d'utilité.
    - *social* : quand la situation l'exige, un agent doit interagir avec les autres agents (logiciels et humains) afin de compléter ses tâches ou aider les autres agents à accomplir les leurs.

De ce qui précède, un agent doit réagir aux changements survenus dans son environnement. Pour ce faire, il procède en trois étapes, selon le schéma suivant *(i)* perception de l'environnement, *(ii)* traitement de données collectées et enfin, *(iii)* réalisation d'actions. Une réaction peut être réalisée soit sur l'état interne de l'agent, soit sur l'environnement. Elle représente une réponse à un stimulus. Ce dernier peut être issu de l'intérieur, suite à des changements dans l'état interne de l'agent lui-même, ou bien de l'extérieur de l'agent issu de changements dans son environnement [13].

2.4.2.2. Types d'agents

Un agent peut être implémenté de plusieurs manières. Peu importe l'architecture adoptée, un agent est vu comme une fonction liant ses perceptions à ses actions. Ce qui différencie les différentes architectures d'agents, c'est la manière dont les perceptions sont liées aux actions. Selon le comportement adopté par un agent, nous pouvons distinguer trois modèles :

2.4.2.2.1. Agent réactif

Ce type d'agents ne fait ni délibération ni planification. Un agent réactif réagit simplement, à sa perception de son environnement, en appliquant certaines règles prédéfinies. Étant donné qu'il n'y a pratiquement pas de raisonnement, ces agents, utilisant un principe de réflexe, peuvent agir et réagir très rapidement [34].

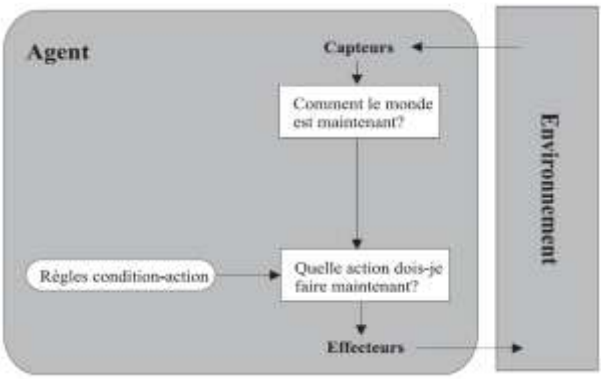


Figure 19. Schéma de fonctionnement d'un agent réactif [50].

2.4.2.2.2. Agent délibératif

Une certaine délibération pour sélectionner ses actions est faite par ce type d'agents, selon le plan *Perception, Planification et Action*. Une telle délibération peut prendre la forme d'un plan qui reflète la liste d'actions que l'agent doit effectuer en vue de réaliser ses objectifs. Ce modèle permet à un agent d'analyser sa situation, anticiper et, ensuite, planifier une action. Ce type d'agents endure l'inconvénient de vitesse et flexibilité limitées. Il est, toutefois, inadéquat pour gérer des évènements inattendus [13].

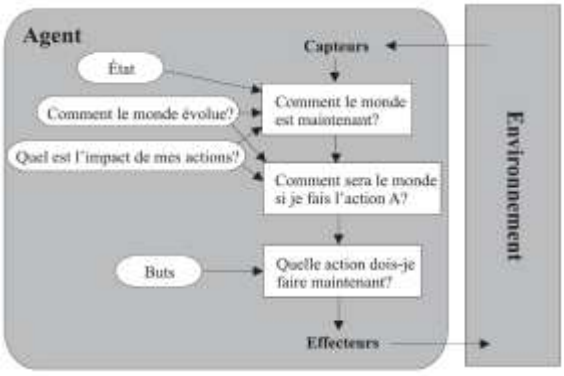


Figure 20. Schéma de fonctionnement d'un agent délibératif [50].

4.2.2.3. Agent hybride

Cette approche fait réunir les deux précédentes, un comportement réactif de base avec un contrôle de niveau supérieur de connaissance. La finalité est d'associer la réactivité des agents avec l'habilité de planification des systèmes cognitifs [9] [11].

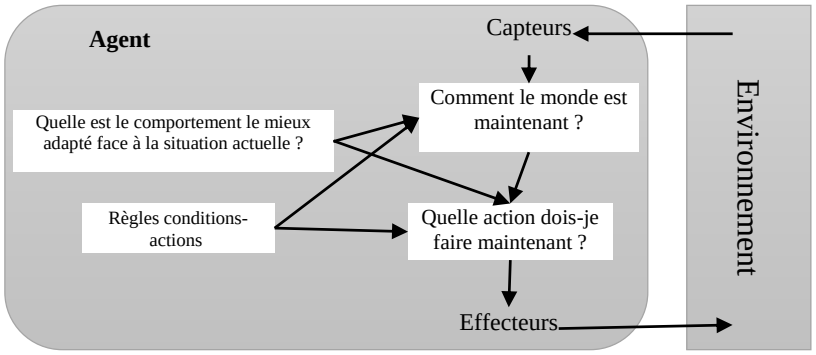


Figure 21. Schéma synoptique de fonctionnement d'un agent hybride.

2.4.2.3. Système multi-agents (SMA)

2.4.2.3.1. Définition d'un *SMA*

[11] définit un *Système multi-agents* (*SMA*) comme étant *un système distribué conçu et implémenté comme un ensemble d'agents interagissant, le plus souvent, selon des modes de coopération, de concurrence ou de coexistence*. La coopération dans un *SMA* peut être vue comme la réalisation d'une interaction, au-delà d'un simple échange de messages, telle que la négociation pour trouver une position commune, une solution globale ou la distribution de tâches. Un SMA possède les caractéristiques suivantes [34] :

- Chaque agent a un point de vue partiel du problème global, donc ses informations et ses capacités de résolution de problèmes sont limitées.
- Il n'y a pas de contrôle global du système.
- Les données sont décentralisées.
- Les calculs sont asynchrones.

Un *SMA* a pour objectif de rassembler un ensemble d'agents et les organiser pour accomplir des objectifs de niveau élevé. Il existe cependant plusieurs types de *SMA*. Nous pouvons citer les systèmes réactifs qui essayent de réaliser un comportement émergent, en se basant sur plusieurs agents, pour résoudre un problème global. Dans un *SMA* et pour garantir la coopération entre ses agents, le besoin d'assurer une communication entre ces agents est essentiel [8].

2.4.2.3.2. Résolution de problèmes avec les *SMA*

Pour résoudre un problème complexe, un *SMA* réactif adapte un comportement émergent en faisant appel à plusieurs entités. Chacune de ces entités possède une connaissance et une capacité d'action limitées. Un *SMA* de niveau supérieur est généralement utilisé en robotique mobile [38]. Dans la hiérarchique d'un *SMA*, une entité de niveau supérieur doit être envisagée pour superviser l'accomplissement d'une tâche. Cette entité sera aussi chargée de centraliser les décisions du système et la communication des messages entre les différents agents constituant le système. Un exemple d'application de ce type de système (*SMA* réactif) est dans la coordination entre plusieurs robots mobiles dans l'opération de transport de conteneurs sur un quai [4]. Toutefois, différentes applications avec ce modèle peuvent être envisagées qui comportent la tâche de génération de trajectoires, associée à un module d'évitement d'obstacles tout en bénéficiant des techniques bio-inspirées d'intelligence artificielle [52]. Les réseaux de neurones et les règles de la logique floue, associés à un *SMA*, peuvent offrir un contrôle de haut niveau pour des systèmes complexes [25].

2.4.2.3.3. Avantages des *SMA*

Les *SMA* procurent une façon facile et efficace de modéliser les systèmes qui nécessitent l'utilisation de plusieurs entités et qui peuvent être géographiquement distribués. Même si une application particulière ne requiert pas l'utilisation d'un *SMA*, il existe tout de même plusieurs raisons pour utiliser ce type de systèmes. Par exemple, ils peuvent s'avérer bien utiles pour des problèmes possédant de multiples méthodes de résolution, de multiples perspectives et/ou de multiples résolveurs.

Dans un système informatique, choisir les *SMA* permet de bénéficier des avantages traditionnels de la résolution distribuée et concurrente de problèmes [11] :

- *Modularité :* cette propriété permet de rendre la programmation plus simple. Elle accroit également les possibilités d'extensibilité, vu que l'ajout de nouveaux agents à un *SMA* est beaucoup plus simple que d'ajouter de nouvelles capacités à un système monolithique.
- *Fiabilité :* une certaine fiabilité peut être également garantie, dans la mesure où le contrôle et les responsabilités étant partagés entre les différents agents, le système peut tolérer la défaillance d'un ou de plusieurs agents. Si une seule entité contrôle tout, alors une seule défaillance de cette entité fera en sorte que tout le système tombera en panne.
- *Vitesse* : le gain en vitesse est dû principalement au parallélisme lors de l'exécution, étant donné que, pour résoudre un problème, plusieurs agents peuvent coexister et travailler en même temps dans un même système.

Les principaux avantages de l'utilisation de l'approche multi-agents par rapport aux autres architectures sont donnés comme suit [58] :

- *Amélioration de la composabilité* : cet avantage permet à l'architecture de contrôle d'être construite en assemblant différents agents.
- *Haute scalabilité (évolutivité)* : cette caractéristique permet d'ajouter de nouvelles fonctionnalités et de nouveaux nœuds à l'architecture de contrôle.
- *Plus grande flexibilité* : cette flexibilité permet de reconfigurer facilement l'architecture.
- *Meilleure maintenabilité* : elle est assurée par la modularité et la facilité de remplacement des nœuds.
- *Fiabilité et tolérance aux pannes* : les architectures multi-agents offrent des avantages en termes de fiabilité et de tolérance aux pannes.

2.5. Contrôle basé sur les systèmes multi-agents

Dans cette section, nous allons présenter, selon un ordre chronologique, quelques travaux de recherche basés sur des systèmes multi-agents ayant proposés des modèles de contrôle des manipulateurs mobiles. Les approches proposées reposent sur un système composé de plusieurs agents qui coopèrent entre eux pour répondre à un besoin de contrôle. Nous essayons à travers cet ordonnancement de suivre l'évolution de l'utilisation des systèmes multi-agents pour le contrôle de ce type de robots.

2.5.1. Contrôle d'un système face à des perturbations

L'utilisation de plusieurs agents a permis au système de contrôle proposé dans [1], dans sa globalité, d'être dirigé avec différents modules pour gérer les tâches, l'environnement ou l'état du système. Dans un tel système, chaque agent doit proposer des actions, tout en connaissant les données d'entrée, en se basant sur son autonomie et ses informations locales propres à lui. Les propositions des agents sont, ensuite, évaluées selon plusieurs critères pour choisir l'une de ces propositions, qui sera, enfin, exécutée par le système.

Cette approche était proposée pour le contrôle d'un système robotisé en présence d'une perturbation par une force humaine. Les concepts proposés sont appliqués sur un système de contrôle pour un manipulateur mobile lors de la réalisation d'une tâche de délivrance d'un objet à un humain.

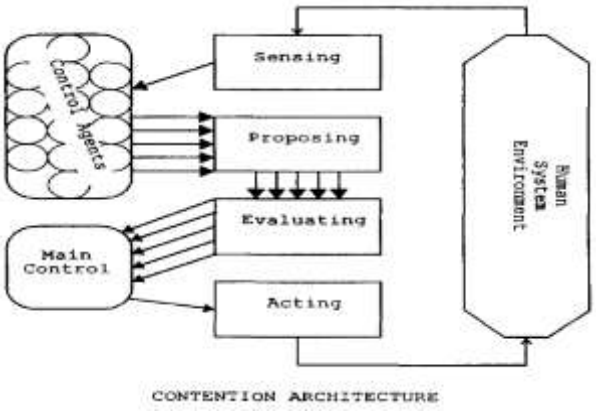


Figure 22. Diagramme de l'architecture de concurrence [1].

Ce modèle rappelle une architecture de contrôle nouvelle qui repose sur les techniques du *soft-computing* avec plusieurs agents distribués. Ces agent coopèrent entre eux pour fournir les informations de sorties (consignes de contrôle), à partir des informations d'entrées (issues de capteurs).

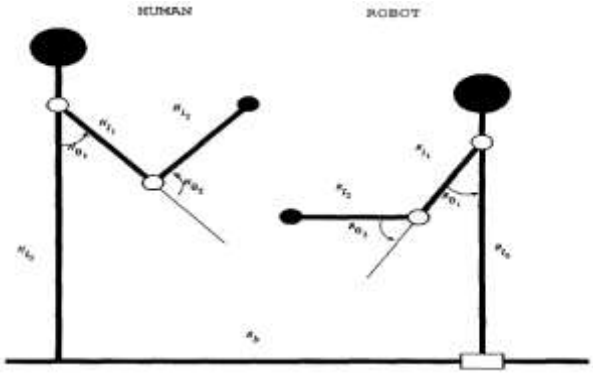


Figure 23. Schéma d'interaction entre le robot et l'humain [1].

2.5.2. Tâche de délivrance d'un objet

Ce modèle de contrôle distribué est basé sur un *SMA*, dont le but est le contrôle d'un manipulateur mobile qui agit avec les opérateurs durant une tâche de délivrance d'objets en deux dimensions. Plusieurs entités, en occurrence les agents, coopèrent les uns avec les autres pour atteindre leurs propres objectifs. L'ensemble de ces objectifs, synthétisés, constituent l'objectif final du système. Les opérations à accomplir pour atteindre l'objectif global sont réparties entre les agents où chacun est responsable d'accomplir son propre opération.

L'approche proposée assure un contrôle en vitesse. Les trois agents identifiés sont implémenté en utilisant un contrôleur flou. Ils sont définis comme suit :

- *Base agent* : ce premier agent contrôle la vitesse de déplacement de la base mobile du robot en fonction des distances horizontales et verticales d_x et d_y entre l'effecteur du robot et celle de l'opérateur.

- *Shoulder agent* : cet agent contrôle la vitesse angulaire de l'épaule en fonction des distances d_x, d_y et de l'angle β. Afin de calculer cet angle, l'agent *Shoulder* a besoin des informations sur l'état de l'agent *Elbow*.
- *Elbow agent* : il contrôle la vitesse angulaire du coude en fonction des distances d_x, d_y et de l'angle α.

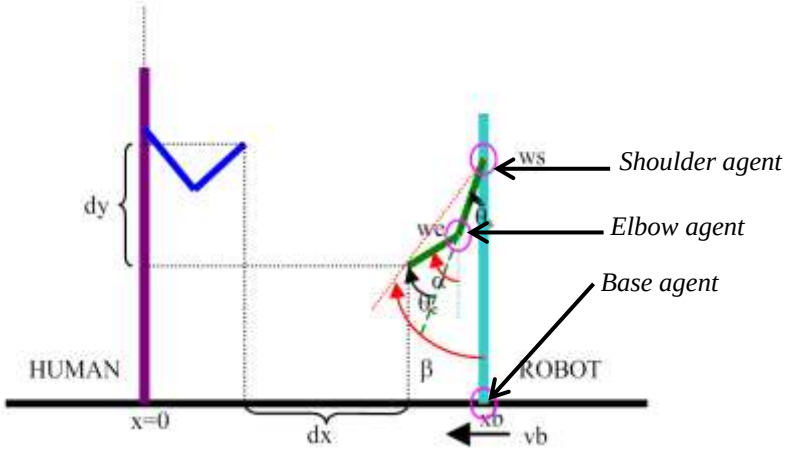


Figure 24. Différents agents du système [59].

Cette approche est basée sur l'idée de considérer l'apport du mouvement propre d'un seul agent en considérant les autres agents comme stationnaires. Cette approche simplifie considérablement la conception et l'implémentation du contrôleur. Dans ce travail, en plus d'utiliser la logique floue pour la partie cognitive, une amélioration additionnelle du contrôle est effectuée en utilisant les algorithmes génétiques. Cela est réalisée, en considérant certains critères, qui s'appuient sur l'ajustement des fonctions d'appartenance des contrôleurs flous. Ces critères sont calculés par l'utilisation d'une fonction *fitness* qui est une combinaison entre *(i)* la distance parcourue par l'effecteur, *(ii)* la durée du manœuvre et *(iii)* l'énergie.

2.5.3. Contrôle intelligent en deux niveaux

Ce modèle avait comme but d'assurer une planification efficace des mouvements pour des manipulateurs industriels, avec un ddl élevé, dans un environnement statique ou dynamiques. Tout en s'appuyant sur une approche multi-agents et une structure hiérarchique, un raisonnement basé sur la logique floue était adopté dans le planificateur de chemin. La structure hiérarchique du système comporte deux niveaux :

- *Un niveau supérieur* : responsable de sélectionner, pour chaque articulation, d'une manière dynamique, un comportement approprié parmi les trois comportements proposés *(i) Slide, (ii) Goal approaching et (iii) Obstacle avoidance*.

- *Un niveau inférieur dans la hiérarchie* : conçu pour déterminer la vitesse articulaire, pour chaque articulation, selon le comportement attribué par le niveau supérieur.

Dans ce modèle, l'utilisation du paradigme multi-agents a offert une flexibilité et une efficacité dans la planification de mouvements pour les manipulateurs avec un *ddl* élevé, sans avoir recours à l'étude des modèles mathématiques. Ce modèle proposé est applicable, dans un contexte temps réel, pour des tâches de planification de mouvements des manipulateurs. Toutefois, dans des scénarios de manipulation complexe, un problème de minima locaux peut survenir. Pour éviter ce problème, les concepteurs de cette approche se sont basés sur un mécanisme de recouvrement appelé *back-tracking*. Malgré le fait que cette conception soit destinée à un environnement dynamique, ce modèle ne considère qu'un seul obstacle. De plus, le critère d'efficacité n'est pas considéré.

2.5.4. Tolérance aux pannes dans la robotique de service

Les auteurs de cette solution, présentée dans [57] et [13], ont construit un système de contrôle, adapté à la robotique de service, autour d'une architecture multi-agents. La solution exposée est destinée aux personnes présentant une déficience motrice grave, pour compenser leur handicap. Cette approche est composée de deux parties (i) une première partie concerne le manipulateur et (ii) une deuxième partie pour la base mobile. Cette architecture a exploité la redondance du robot pour offrir une tolérance à certaines pannes. L'exploitation de la redondance du système a permis également de montrer des comportements ou des configurations au robot qui facilitent l'appropriation de la machine par la personne et donc la coopération homme-machine.

À chaque articulation du manipulateur est affecté un agent réactif. Les agents du système coexistent et s'exécutent en parallèle dont le but est que chacun réalise une tâche locale, sans qu'il possède une connaissance préalable des actions des autres agents. Un comportement global émergeant est alors attendu, qui consiste à amener le manipulateur vers une position complètement tendu, tout en essayant de se rapprocher au maximum de l'objectif à rejoindre.

2.6. Conclusion

Après avoir donné la définition d'un contrôleur, ce chapitre a présenté, dans un premier lieu, les différentes architectures de contrôle des manipulateurs mobiles, en commençant par les architectures délibératives, ensuite, les architectures comportementales et les architectures hybrides (délibérative et réactive). Dans un deuxième lieu, nous avons exposé une classification en deux approches de contrôle des manipulateurs mobiles *(i)* les approches roboticienne considérées comme générales et classiques, et *(ii)* les approches basées sur les techniques de l'IAD où le contrôle des

robots est essentiellement distribué entre plusieurs agents concurrents, généralement organisé de manière hiérarchique, fusionnant des compétences réactives et délibératives. Les compétences réactives sont généralement simples et doivent être exécutées de façon très rapide. Par contre, les compétences délibératives sont plus complexes et exigent, naturellement, une exécution plus lente.

La méthode générale utilisée pour le contrôle des systèmes robotisés se base sur le calcul des modèles mathématiques. Ces modèles sont essentiels pour le calcul des mouvements et des consignes nécessaires pour l'accomplissement des tâches. Cette approche produit de bons résultats lorsqu'il s'agit d'effectuer des tâches répétitives dans un environnement connu. Mais, elle souffre de l'inconvénient de complexité de calcul, qui doit se faire hors ligne, et du fait de ne pas tolérer un changement dans la structure mécanique du système. Ce dernier ne prend pas en considération le cas où une articulation tombe en panne. Pour éviter et remédier à ce genre de problèmes, l'approche multi-agents offre des méthodes qui font appel au paradigme agent en proposant une décomposition du système robotisé en un ensemble d'agents distincts. Cette dernière approche bénéficie des avantages de la résolution distribuée des problèmes. Elle pose, cependant, le problème de la gestion et du contrôle des agents et de leurs ressources partagées. Une contrainte à ne pas négliger, lors de la conception d'un système de contrôle, est le manque d'information. Ce manque est dû principalement à la marge des erreurs de mesure livrée par les capteurs physiques.

Vers la fin de ce chapitre, nous avons présenté quelques travaux de recherche récents qui ont proposés des modèles intéressants, utilisant une méthodologie multi-agents, pour le contrôle des manipulateurs mobiles.

Le chapitre suivant est dédié à la conception de notre propre approche de contrôle des manipulateurs mobiles. L'approche proposée a comme but de s'adapter à une structure composée de ressources de nature hétérogène (une base mobile, un manipulateur). À cet effet, un *SMA* est chargé de contrôler le système robotisé tout en partageant le contrôle des ressources. À chaque ressource est affectée un ou plusieurs agents.

CHAPITRE 3
APPROCHE DE CONTRÔLE PROPOSÉE

3.1. Introduction

Dans le cycle de vie d'un logiciel, le choix d'une méthodologie de développement des contrôleurs de robot, qui soit appropriée à tous types de manipulateurs mobiles, est un problème complexe qui ne pouvant pas être mené de façon générique [39].

La conception est la phase la plus stratégique, il est cependant nécessaire de comprendre et d'identifier les besoins de l'architecture afin de la concevoir et de l'implémenter d'une manière correcte. Avec l'augmentation de la complexité des architectures à élaborer, l'utilisation d'une méthodologie de développement apparaît primordiale. Néanmoins, l'absence de telles méthodes couvrant l'ensemble du cycle de vie d'un SMA rend la tâche très difficile et très compliquée [29].

Suite aux notions de base présentées dans les chapitres précédents, ce chapitre s'intéresse au développement et à la mise en œuvre pratique de notre propre approche de contrôle des manipulateurs mobiles. Il présentera, en premier lieu, une analyse du système robotique expérimentale pour lequel le contrôleur a été développé, en occurrence le manipulateur mobile *RobuTER/ULM*. En deuxième lieu, nous décrivons l'architecture globale du système, tout en présentant une vue globale d'une nouvelle approche de contrôle des manipulateurs mobiles et, en particulier, *RobuTER/ULM*. Ensuite, nous donnons les détails de modélisation de l'approche, la spécification et la composition des différents agents ainsi que l'échange d'informations entre eux, en

faisant recours aux diagrammes UML qui représentent un outil, à la fois, uniforme et universel de conception des systèmes informatiques. En troisième lieu, les détails d'implémentation et les outils de mise en œuvre utilisés sont présentés. Enfin, nous nous intéressons aux valeurs ajoutées et aux avantages offerts par le mécanisme de contrôle proposé.

3.2. Spécification globale

3.2.1. Système robotique expérimental

L'objectif fondamental d'un contrôleur de robot est de générer les mouvements correspondants aux différentes tâches assignées. À présent, il convient de préciser les exigences fonctionnelles appropriées à notre cas (le manipulateur mobile RobuTER/ULM). RobuTER/ULM se compose d'une base mobile non-holonome avec deux roues motrices et deux roues folles. Sur cette base mobile on a fixé un manipulateur ultraléger à six ddl rotoïdes et comme effecteur une pince électrique à deux doigts, avec un capteur d'effort six axes. Tout autour de la base mobile, une ceinture de 24 capteurs à ultrason ainsi qu'un capteur laser (LMS) dans sa partie frontale sont montés pour la perception de l'environnement et la détection d'obstacles. Ce robot est destiné à accomplir des tâches d'intervention et de manutention dans des environnements contraints.

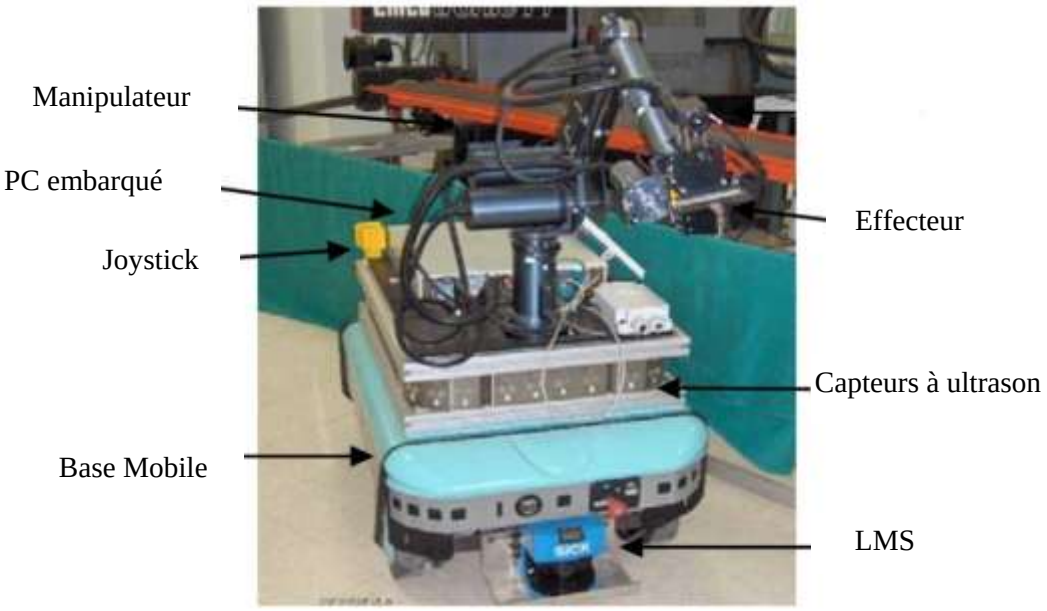


Figure 25. Manipulateur mobile RobuTER/ULM.

Une tâche, que le RobuTER/ULM doit accomplir, est composée d'un ensemble d'actions de mouvements et de manœuvres, qui sont eux-mêmes définies par des séquences de consignes. Exécuter une tâche consiste à interpréter et synchroniser les consignes qui la composent.

3.2.2. Analyse des besoins

Pour notre contrôleur, nous avons adopté une spécification qui se veut être la plus générique possible, tout en étant indépendante des plateformes logicielles utilisées ainsi que des caractéristiques physiques du robot à contrôler (caractéristiques de la base mobile, nombre d'articulations, types d'actionneurs, etc.). La conception et l'implémentation du contrôleur proposé ne fait qu'adapter la dite spécification aux contraintes présentées par le manipulateur mobile expérimental RobuTER/ULM.

Une spécification en *UML* débute par l'analyse des besoins. Cette analyse est menée par les cas d'utilisation à travers lesquels on exprime les besoins fonctionnels du système. La spécification se poursuit par une analyse des fonctionnalités décrites dans les *cas d'utilisation.*

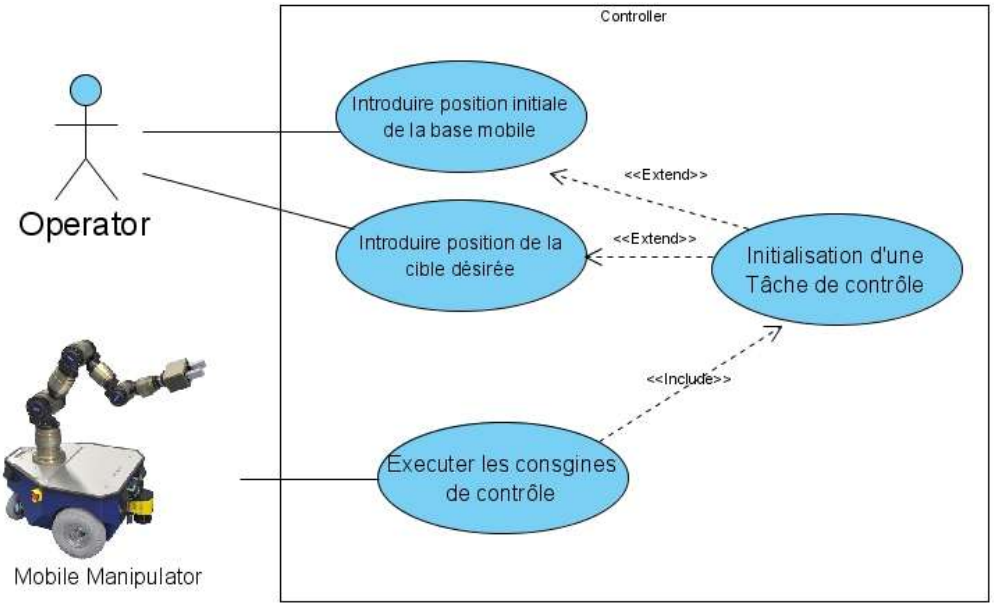


Figure 26. Diagramme de cas d'utilisation pour initialisation d'une tâche de contrôle.

Notre analyse des besoins a pour but la description des services assurés par le système. Elle est basée sur les cas d'utilisation exprimé sous la forme d'interactions entre les acteurs et le système. Dans notre cas de figure, nous comptons un seul cas d'utilisation où deux acteurs sont identifiés *(i)* l'*opérateur* et *(ii)* le *manipulateur mobile*. L'opérateur est l'acteur responsable du lancement d'une tâche, tandis que le manipulateur mobile est un acteur passif qui subit les instructions du contrôleur. La réalisation d'une tâche de contrôle consiste à faire mouvoir l'effecteur du robot d'une situation initiale $Situation_{Init}$ vers la situation cible $Situation_{Finale}$ introduite par l'opérateur. Le diagramme de ce cas d'utilisation est présenté dans la figure précédente.

Les diagrammes de séquence sont des outils adaptés à l'indentification des relations entre les différentes entités (classes) dans un système. Ils offrent la possibilité de représenter les interactions entre les instances (objets) à l'aide des scénarios. Dans ce qui suit, à travers un diagramme de

séquence (figure 27) qui détaille le cas d'utilisation *Initiate Task*, nous montrons les interactions entre le contrôleur et les deux acteurs du système. Le contrôleur est représenté par une boite noire interagissant entre le manipulateur mobile et l'opérateur humain qui entreprend le processus de contrôle. Un processus commence par l'initialisation d'une tâche de contrôle, en introduisant les coordonnées de la cible à atteindre par le robot. Une tâche consiste en un échange de messages, entre le contrôleur et le manipulateur mobile, dans les deux sens. À chaque itération, le contrôleur requiert la situation actuelle du manipulateur mobile (position et orientation de la base mobile, coordonnées articulaires du manipulateur, etc.). Ensuite, le contrôleur calcule et envoie des consignes de contrôle au robot selon sa situation actuelle par rapport à la cible à atteindre. Le processus de contrôle s'arrête, une fois la cible désirée est atteinte, en communiquant un message à l'opérateur.

La situation d'un manipulateur mobile est définie par la position et l'orientation de la base mobile (x_b, y_b, θ_b) ainsi que les coordonnées articulaires du manipulateur (q_1, q_2, ..., q_n). Une cible est considérée comme un point accessible dans l'espace opérationnel du manipulateur mobile. Elle est représentée par les trois coordonnées cartésiennes (x_c, y_c, z_c).

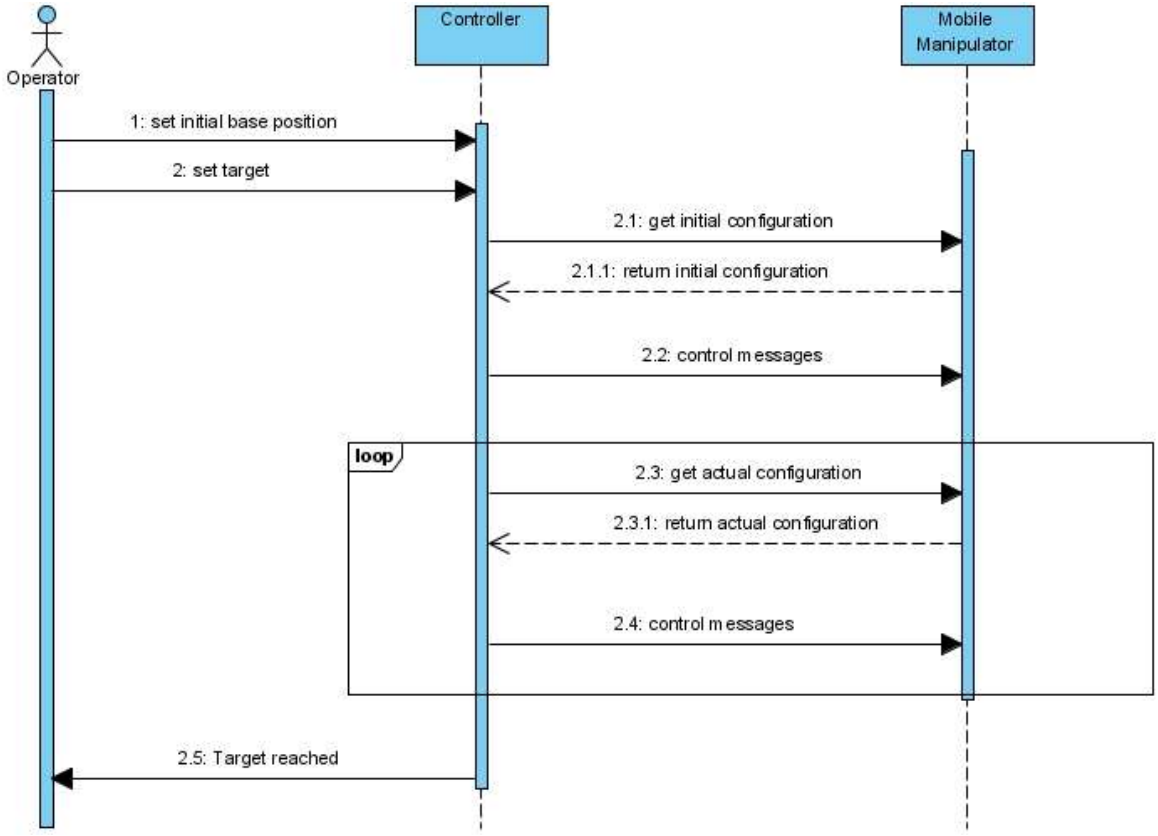


Figure 27. Diagramme de séquence pour lancer la réalisation d'une tâche.

L'approche que nous avons proposée est basée sur le schéma de contrôle décrit dans le diagramme de séquence précédent (Figure 27). Le processus de contrôle se déroule en plusieurs itérations. Dans chaque itération, le contrôleur requiert la situation actuelle du manipulateur mobile

ainsi que celle de la cible pour pouvoir déduire les consignes qui seront transmises au robot. Le processus de contrôle s'achève une fois l'effecteur atteint la position cible. Un message est alors transmit pour informer l'opérateur.

Le schéma synoptique de la figure 28 explique le processus de contrôle. Le contrôleur est représenté par une boite noire qui admet comme entrée une erreur (ε_P). En se basant sur la situation du manipulateur mobile (P_R), les coordonnées de l'effecteur sont calculées à l'aide du *MGD*. Le contrôleur renvoie les données de contrôle (consignes en vitesses et/ou en position) au manipulateur mobile. Pour la base mobile, ces consignes sont soient les vitesses de rotation/translation, l'angle de rotation ou la distance à parcourir en translation. En ce qui concerne le manipulateur, les consignes de contrôle seront les vitesses articulaires ou bien les angles de rotation pour chaque articulation.

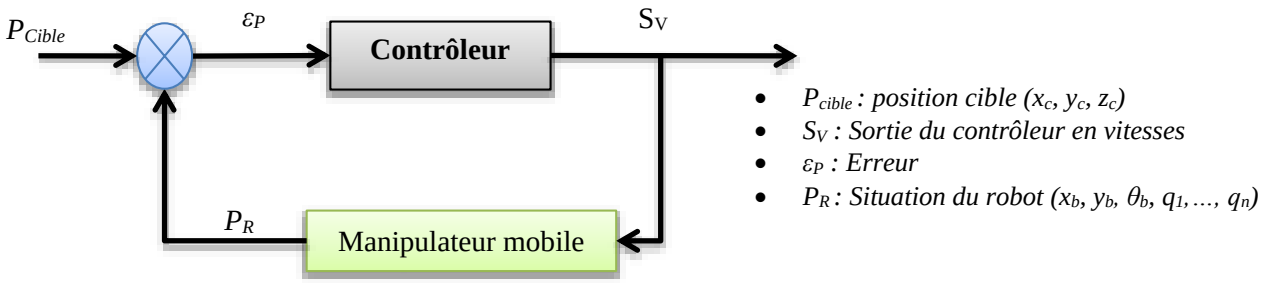


Figure 28. Schéma synoptique pour le contrôleur du manipulateur mobile.

À ce stade, nous avons considéré le contrôleur comme une boite noire qui opère, en réponse à des changements dans l'état du système, sur le manipulateur mobile. Dans ce qui suit, nous allons présenter une analyse détaillée sur les vues fonctionnelles du schéma de contrôle. Cette proposition est un schéma générique et qui pourra, par la suite, être adapté aux spécificités liés aux différents manipulateurs mobiles.

3.2.3. Spécification conceptuelle

Le manipulateur mobile *RobuTER/ULM* est appelé à accomplir des tâches d'intervention dans un environnement contraint. Pour ce type de robots, une tâche d'intervention sollicite, à la fois, les capacités de manipulation du manipulateur ainsi que la capacité de locomotion de la base mobile. La réalisation d'une tâche se résume alors au déplacement de l'effecteur du robot à partir d'une situation initiale vers une situation cible finale. Pour effectuer ce déplacement, le manipulateur ainsi que la base mobile sont amenés à coordonner leurs mouvements.

En s'inspirant des travaux de recherche présentés dans la partie état de l'art, nous avons proposé un nouveau modèle pour le contrôle des manipulateurs mobiles. Ce modèle se repose sur une approche multi-agent, où chaque agent du système est une entité à part entière qui possède des objectifs propres. Un agent est doté d'une certaine autonomie dans la prise de décision tout en étant

capable de coordonner avec les autres agents du système pour réussir un comportement émergent. Ce comportement global se manifeste à travers la collaboration, par le biais de messages échangés, entre les différents agents. Pour réussir ce comportement émergent, un agent est affecté à chaque articulation du manipulateur ainsi qu'un autre agent pour le contrôle de la base mobile. Chacun de ces agents requiert la position actuelle de l'effecteur ainsi que celle de la cible. Les coordonnées cartésiennes de ces deux points sont utilisées lors de l'estimation des erreurs. Dans ce travail, nous n'avons considéré que l'erreur en distance et en orientation. Le déplacement entrepris par chaque agent est régulé selon la variation de ces erreurs. Le schéma ci-dessous montre la disposition de l'erreur en distance ainsi que les projections des erreurs angulaires pour un manipulateur avec trois axes rotatifs.

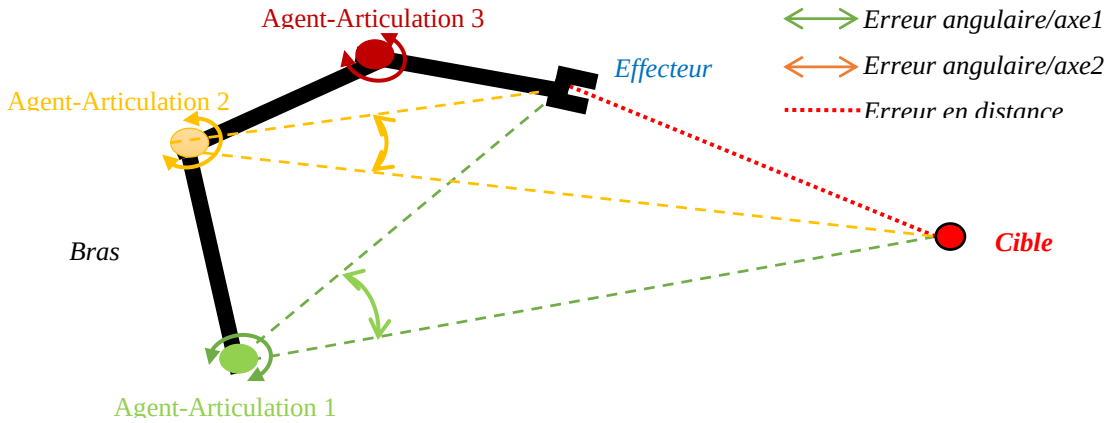


Figure 29. Affectation des agents pour un manipulateur à trois axes rotatifs.

Lors de la réalisation du déplacement de l'effecteur d'une position initiale quelconque vers une position cible, la nature des opérations effectuées par les différents agents varie, d'une itération à l'autre, selon l'évolution de l'effecteur par rapport à la cible. Alors les écarts de distances et d'orientations, entre les coordonnées actuelles de l'effecteur et celles de la position cible, seront utilisées comme entrées de contrôleur.

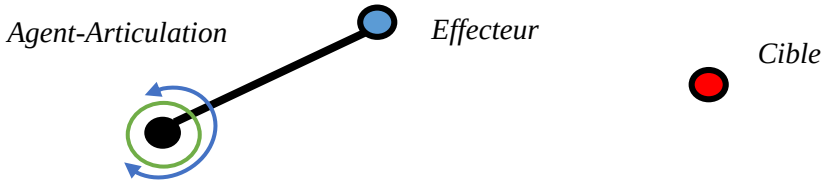


Figure 30. Schéma d'ensemble (Agent-Articulation, Effecteur et Cible).

Les entrées de contrôle peuvent être des consignes en vitesse, comme le montre la figure ci-dessous, ou bien, des consignes en positions :

- les angles de rotation pour les articulations du manipulateur.
- l'angle de rotation ou distance de translation pour la base mobile.

La *Figure 31* montre une proposition pour un plan de contrôle en vitesse, où nous avons affecté à chaque articulation du manipulateur un seul agent réactif *(A_{axe1},…, A_{axen})* ainsi qu'un autre agent hybride *(A_{Base})* pour le contrôle de la base mobile. La sortie d'un agent de contrôle est une consigne en vitesse pour le manipulateur mobile. Les agents assignés aux différentes articulations ont des vitesses articulaires *(V_1, …, V_n)* comme sorties. Pour l'agent de la base mobile, nous comptons deux sorties, une vitesse de rotation (V_R) ainsi qu'une vitesse de translation (V_T).

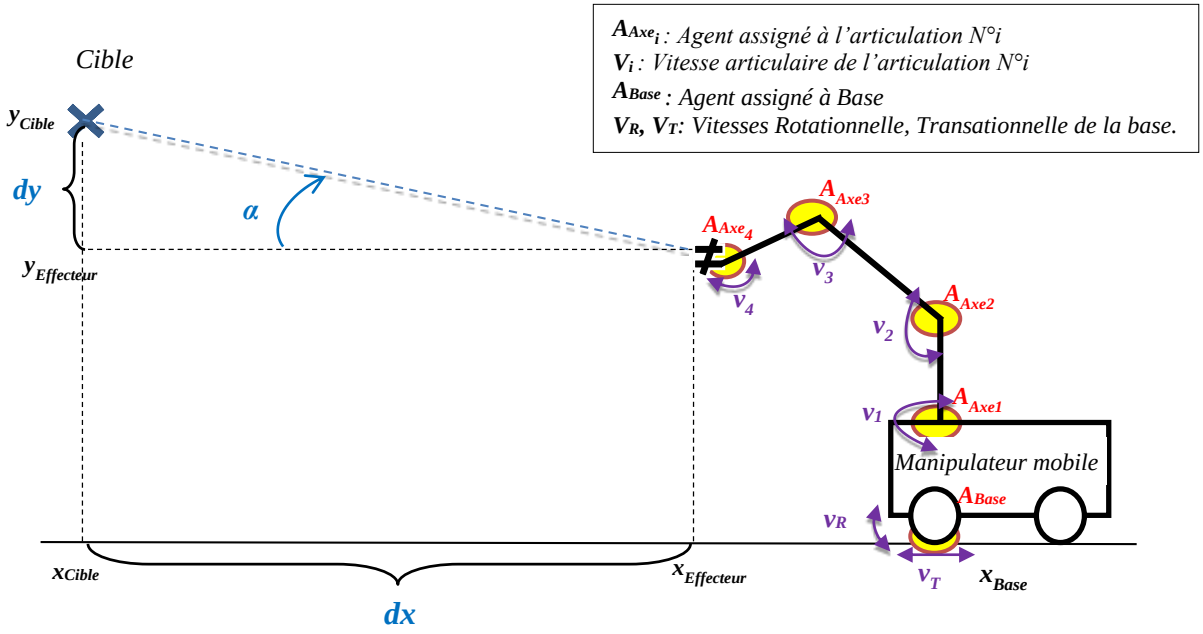


Figure 31. Schéma de contrôle en vitesse pour un manipulateur mobile.

3.2.4. Analyse fonctionnelle

Comme nous l'avons déjà mentionné, chaque sous-système est contrôlé par un ou plusieurs agents. Ainsi, nous affectons pour chaque articulation du manipulateur *ULM* un agent avec un comportement réactif. En ce qui concerne la base mobile, un agent hybride assure le contrôle de ce sous-système. Le contrôle que nous adoptons est un contrôle en position, c'est-à-dire, les sorties envoyées par chaque agent sont des consignes de position. Leur nature est comme suit :

- *Pour les articulations* : ce sont les angles et la direction de rotation (*Up*, *Down*) à partir de la position actuelle.
- *Pour la base mobile* : nous distinguons deux mouvements possibles (*i*) mouvement en rotation, (*ii*) mouvement en translation. De ce fait, les consignes sont respectivement (*i*) les angles et la direction (*Left* ou bien *Right*) de rotation, ou bien (*ii*) la distance et la direction de translation (*Forward* ou *Backward*).

Cette approche, tout en étant simple à concevoir, nécessite plusieurs contraintes. Chaque agent est indépendant dans le processus de prise de décision. Pour la détermination de sa proposition, un agent considère que les autres agents, contrôlant les autres parties du robot, soient stationnaires. Dans le schéma ci-dessous, nous faisons appel à cette affectation pour le cas du manipulateur *ULM* du *RobuTER/ULM*, où chaque articulation est contrôlée par un agent réactif.

Chacun de ces agents essayera de faire coïncider la position de l'effecteur du manipulateur avec celle de la cible. L'apport du mouvement élémentaire pour chaque articulation est évalué. Sa valeur sera, ensuite, comparée avec toutes les autres articulations ainsi que l'apport du mouvement élémentaire de la base mobile. Une entité supérieure est, alors, chargée de sélectionner le meilleur mouvement à entreprendre. Cette entité est implémenté via un agent hybride, appelé *Agent Superviseur*. Le meilleur mouvement est celui qui minimise l'erreur (ε_P). Dans notre cas, celui qui minimise la distance séparant la position cartésienne de l'effecteur et celle de la cible à atteindre.

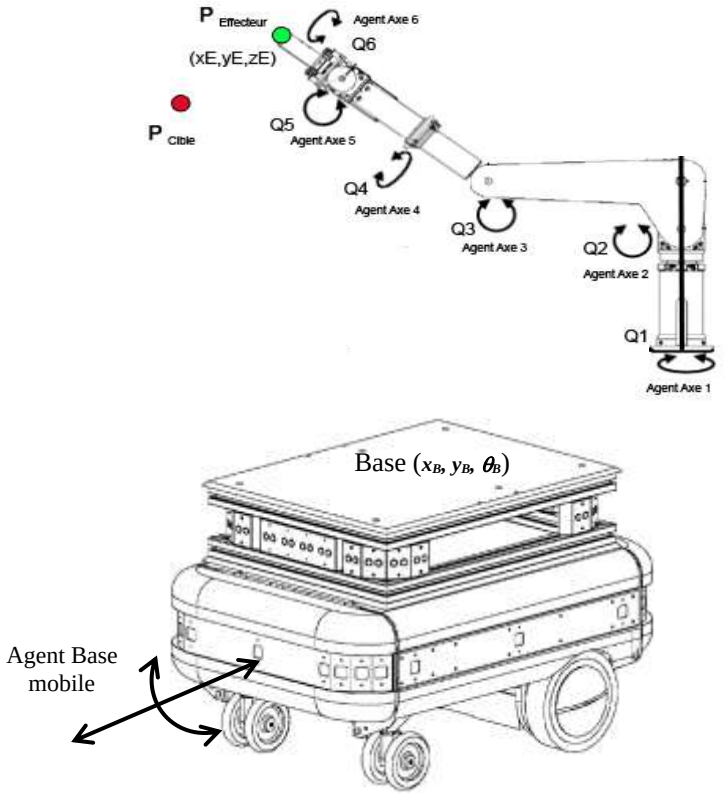


Figure 32. Affectation des agents de contrôle pour le RobuTER/ULM.

3.3. Conception du contrôleur

Dans cette partie, nous allons reprendre en détail le schéma de contrôle présenté précédemment. De plus, le modèle défini sera adapté à l'environnement du développement tout en respectant la structure du modèle et en tenant compte des caractéristiques et des exigences restrictives du système *(i)* type de la base mobile, *(ii)* nombre d'articulations, *(iii)* type de capteurs, *(iv)* contraintes sur l'environnement cible, etc.).

3.3.1. Vues et compositions logiques du système

Le contrôle d'un manipulateur mobile nécessite l'intervention d'une ou plusieurs entités différentes. L'ensemble des interactions entre ces dernières, identifiées lors de la spécification, permet d'obtenir un diagramme de classes global. Dans ce diagramme, une vue générale des différents acteurs du système de contrôle est schématisée.

La meilleure façon d'aborder un système complexe consiste à le décomposer en sous-systèmes élémentaires. Pour accroitre son évolutivité, nous avons opté pour une structure basée sur les paquetages[3] (*Package*) et nous avons structuré notre système selon une organisation modulaire et hiérarchique. L'utilisation de paquetages, pour regrouper les modules et les entités ayants des liens communs, offre une séparation qui est, à la fois, logique et intuitive dans le sens où, cette séparation permet de répartir les fonctionnalités d'un système selon ses structures matérielles ou logicielles.

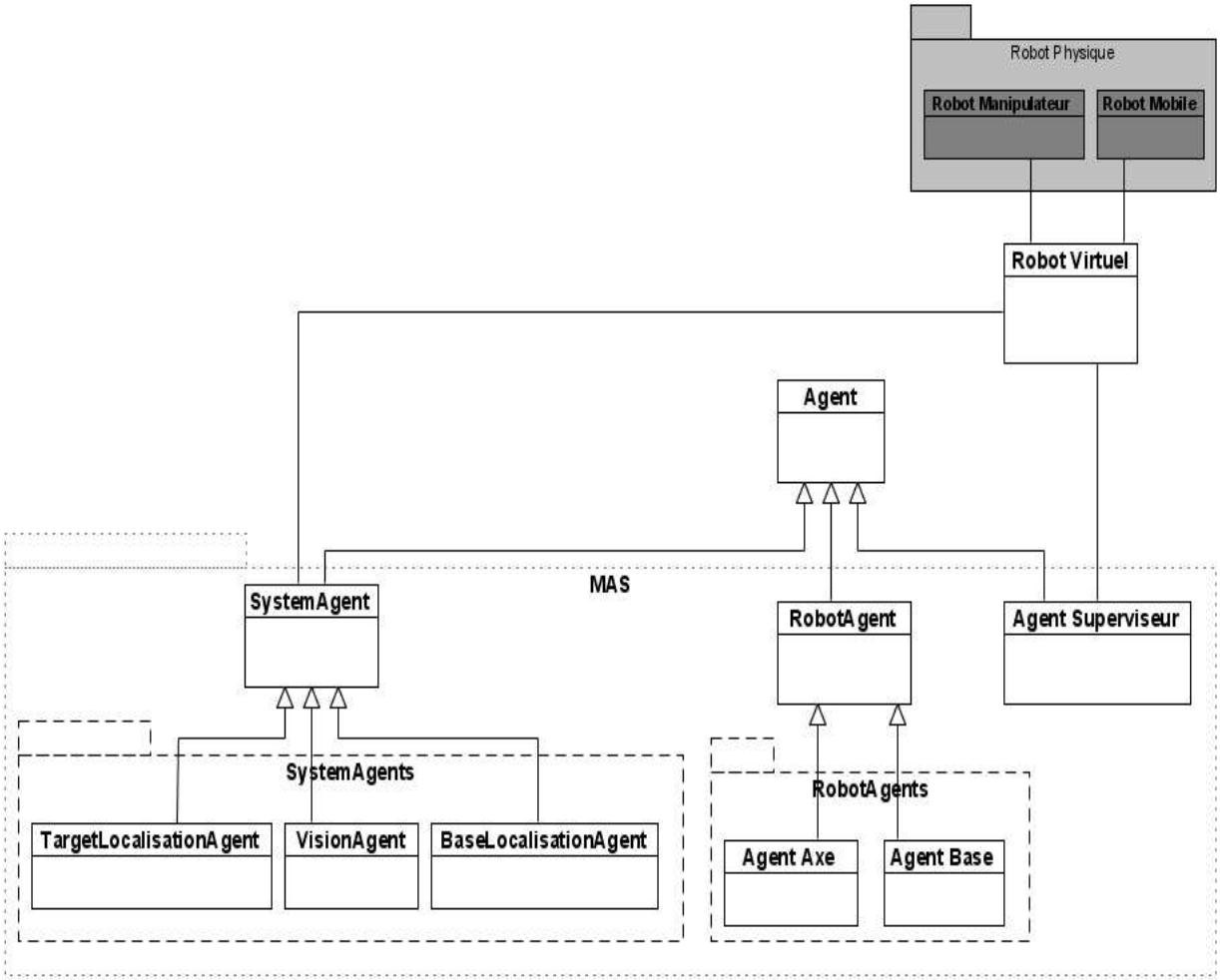


Figure 33. Diagramme de classes global pour le système de contrôle

[3] Un paquetage permet de mettre en œuvre un découpage suivant les différentes vues architecturales et fonctionnelles qui constituent un système donné.

La figure suivante (Figure 33) montre le diagramme de classes du système proposé dans sa globalité. Nous distinguons plusieurs vues présentées par les différents paquetages du système. Chaque paquetage regroupe les classes qui assurent des fonctionnalités communes où celles qui possèdent un lien logique (héritage de la même classe mère où assurer un rôle ou une activité similaire). Les fonctionnalités du système et sa partie opérationnelle sont regroupées dans le paquetage *MAS* (*Multi-agent system*).

Le paquetage *MAS* est le module qui représente l'architecture multi-agents. Il regroupe tous les agents du *SMA*. Ces derniers héritent tous de la même classe, en occurrence la classe *Agent*, où un agent est défini comme étant une entité à part entière qui sera créée au lancement du système. Un agent possède son propre identifiant, un plan de fonctionnement dédié et des objectifs propres à lui. Cependant, selon leur vocations et la nature de leurs traitements, deux types d'agents sont identifiés *(i)* les *agents système*, et *(ii)* les *agents de contrôle.*

3.3.2. Agents système

Les agents de ce type héritent de la classe *SystemAgent*. Ils appartiennent tous au même sous-paquetage *SystemAgents*. Ils sont chargés d'assurer le traitement des données issues des différents capteurs possibles. Ce type d'agents peut regrouper plusieurs fonctionnalités qui ont un lien direct avec la collecte de données qui rentrent dans le processus de contrôle. Les fonctionnalités que nous avons proposé sont *(i)* un module de vision, *(ii)* un module de localisation du robot dans son environnement, et *(iii)* un module pour la localisation d'une cible en mouvement. Toutefois, cette liste n'est pas exhaustive, d'autres fonctionnalités peuvent être intégrées, au fur et à mesure, dans le processus de contrôle.

3.3.3. Agents de contrôle

Ils héritent de la classe *RobotAgent*. Les agents de ce type sont regroupés dans le paquetage *RobotAgents*. Un agent de contrôle, comme son nom l'indique, est dédié à l'opération de contrôle proprement dite. À chaque agent est affecté la tâche de contrôler un sous-système mécanique, qui peut être soit (*i*) la base mobile soit (*ii*) l'une des articulations du manipulateur. Nous avons, donc, identifié un seul agent, *Agent-Base*, pour le contrôle de la base mobile et, un agent *Agent-Axe* est affecté pour le contrôle de chaque articulation. Chacun de ces agents est chargé de générer les consignes de contrôle qui seront envoyées au robot en passant par l'*Agent Superviseur*.

L'objectif de chaque agent de contrôle est d'optimiser la fonction objective donnée par la formule suivante :

$$f_{Obj} = F\_Objective(Configuration, Base, Cible) \quad (1)$$

telles que :

- *Configuration est définie par* ($q_1, \ldots, q_n$) : elle correspond à la configuration courante du manipulateur. Chaque q_i représente la valeur actuelle en degrés de la variable articulaire (*n* est le nombre d'articulations).
- *Base est définie par* (x_B, y_B, θ_B) *:* c'est la situation courante (position et orientation) de la base mobile.
- *Cible est définie par* (x_C, y_C, z_C) : elle représente la situation finale que l'effecteur doit atteindre.

Dans ce travail, l'objectif est de minimiser la distance entre la position de l'effecteur et celle de la cible imposée. Cette distance est calculée comme suit :

$$Distance = \sqrt{(x_{Cible} - x_{Effecteur})^2 + (y_{Cible} - y_{Effecteur})^2 + (z_{Cible} - z_{EEffecteur})^2} \quad (2)$$

Pour calculer la position actuelle de l'effecteur, définie par *Effecteur*($x_E, y_E, z_E, \psi_E, \theta_E, \varphi_E$), on doit passer par le *MGD* du robot. Ceci est réalisé en fonction de *Base*(x_B, y_B, θ_B) et *Configuration* ($q_1, \ldots, q_n$) via l'équation suivante :

Effecteur = MGD(Base, Configuration) (3)

Chaque agent de contrôle (Agents *Axes* et Agent *base mobile*) reçoit, à partir de l'agent *Superviseur*, les informations suivantes :

- Les situations initiales de la base mobile donnée par $Base_{Init}(x_B, y_B, \theta_B)_{Init}$ et du manipulateur donnée par $Configuration_{Init}(q_1, \ldots, q_{ddl})_{Init}$.
- La situation finale à atteindre *Cible*(x_C, y_C, z_C).
- La valeur initiale de la fonction objective f_{Obj}, qui représente la valeur initiale de l'erreur de positionnement entre les deux positions $Effecteur_{Init}$ et *Cible*.

3.3.3.1. Agents Axes

Les agents de ce type possèdent un comportement réactif. Ils réagissent en conséquence pour répondre à des stimuli externes ou internes. Tout en présentant un comportement cognitif limité, un *agent axe* ne possède qu'un objectif local à atteindre. Le principe de fonctionnement pour chaque agent de ce type est le suivant : Un agent possède deux possibilités de mouvement :

- L'agent fait une rotation virtuelle vers le sens positif avec un pas fixe égale à *Joint_footstep*. Ce mouvement est appelé *MoveUp*.

- L'agent réalise le même mouvement précédent et avec le même pas *Joint_footstep*, mais cette fois-ci dans le sens contraire. Ce mouvement est appelé *MoveDown*.

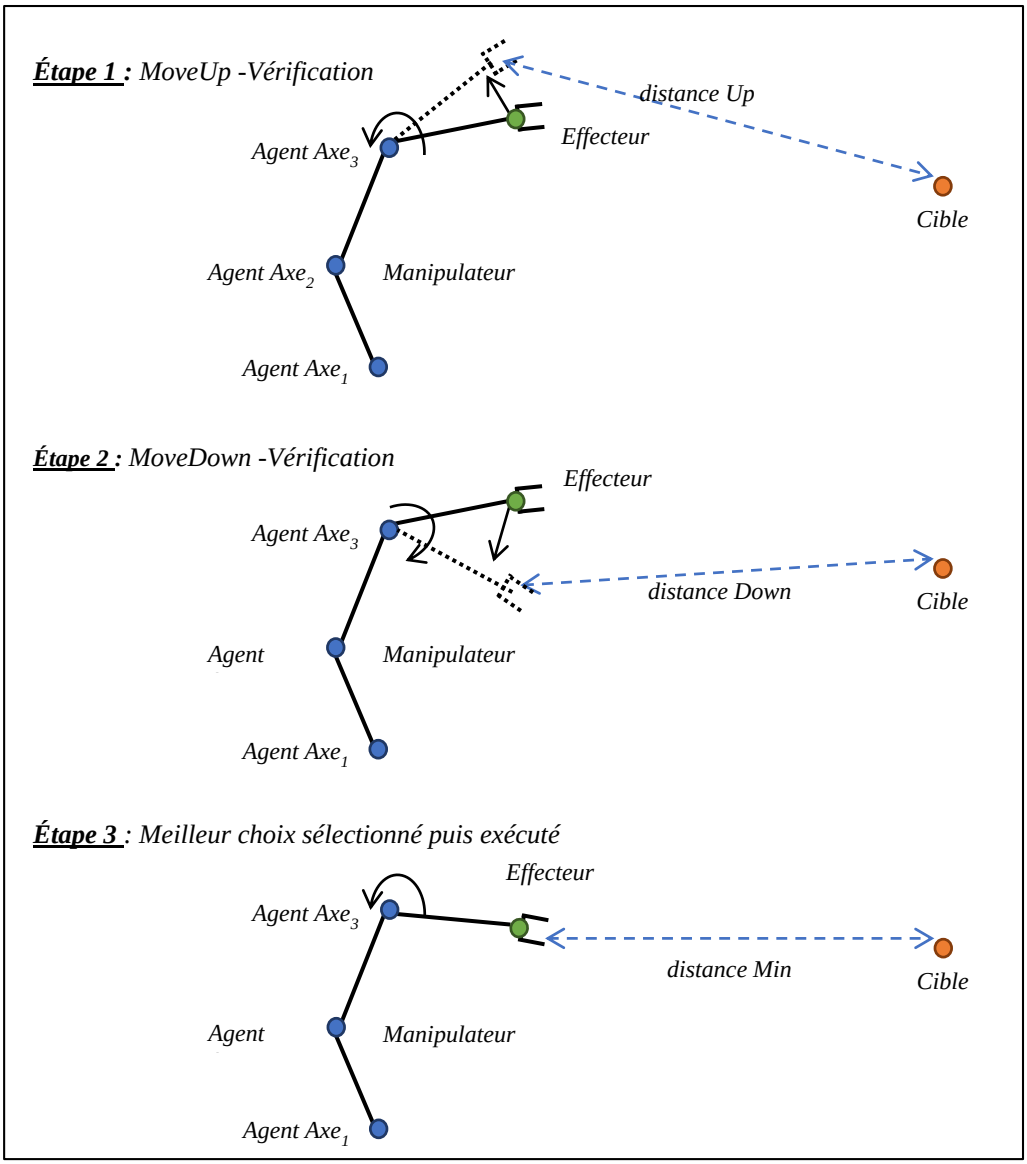


Figure 34. Principe de mouvement virtuel-vérification pour les Agents Axes.

Après chacun des deux mouvements virtuels, l'*Agent-Axe* calcule les deux nouvelles valeurs de la fonction f_{Obj}. Ensuite, il fait une comparaison entre les deux nouvelles valeurs mutuellement et avec la valeur reçue de l'agent *Superviseur*. Le mouvement qui sera sélectionné est celui qui minimise la distance entre *Effecteur* et *Cible*, nous l'appelons le *meilleur choix*. Il faut noter que ce *meilleur choix* peut être, dans certains cas, de rester dans la position initiale. Cela représente le cas où les deux déplacements virtuels augmentent, au lieu de diminuer, la valeur de f_{Obj}.

Après avoir sélectionné le *meilleur choix*, chaque agent *Axe* envoie, à la fin de chaque itération, la valeur qui satisfait le plus son objectif local, à l'*agent superviseur*. Un comportement global émergera de l'association des tous les comportements locaux des *agents axes*. Ce qui permettra, éventuellement de satisfaire l'objectif de niveau supérieur.

La procédure à suivre pour trouver la meilleure proposition est réalisée en trois étapes. Les deux premières étapes auront pour objectif de trouver la position qui rapproche le plus la position de l'effecteur avec celle de la cible désirée. Lors de la troisième étape, on exécute le mouvement sélectionné. Ce principe qui est schématisé par la figure ci-dessus est détaillé formellement dans le diagramme d'activité de la figure suivante.

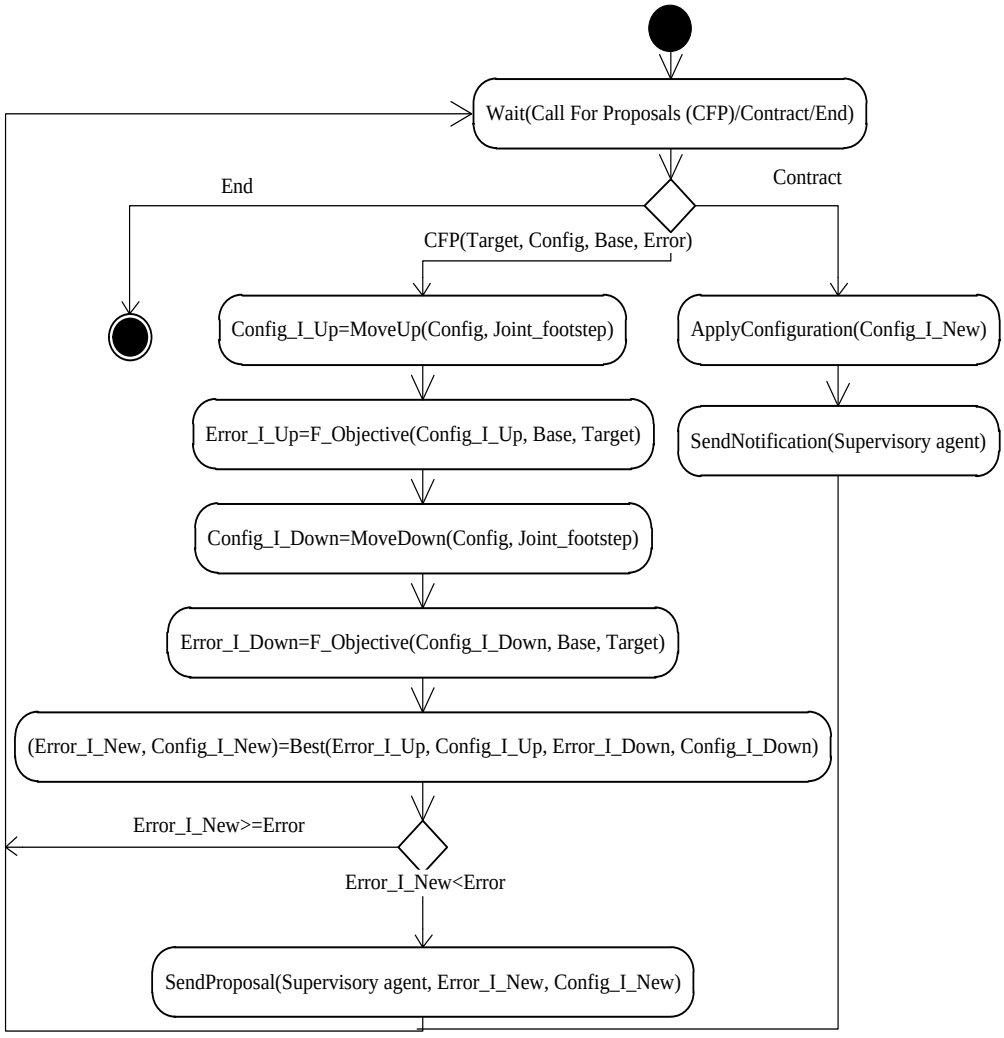


Figure 35. Diagramme d'activité qui détaille le comportement d'un *Agent Axe*.

Après sa création, un *Agent Axe* va se mettre en état d'attente (*wait*) d'un message d'activation issu de l'agent *superviseur*. Selon la nature de ce message, l'*Agent Axe* adaptera son comportement. Un message de type *End* est envoyé pour suspendre l'exécution et arrêter l'agent. Un message de type *CFP* (*Call For Proposal*) a pour objectif de solliciter un agent afin qu'il propose une offre. Le troisième type de message est *Contract*, pour accepter la proposition de l'agent et lancer l'exécution du mouvement proposé.

3.3.3.2. Agent Base mobile

L'environnement où la base mobile évolue est considéré libre de tout obstacle. Toutefois, un mécanisme dédié à l'évitement d'obstacles présents dans l'environnement peut facilement être implémenté au niveau de l'Agent *Base mobile* conjointement avec le schéma de contrôle développé.

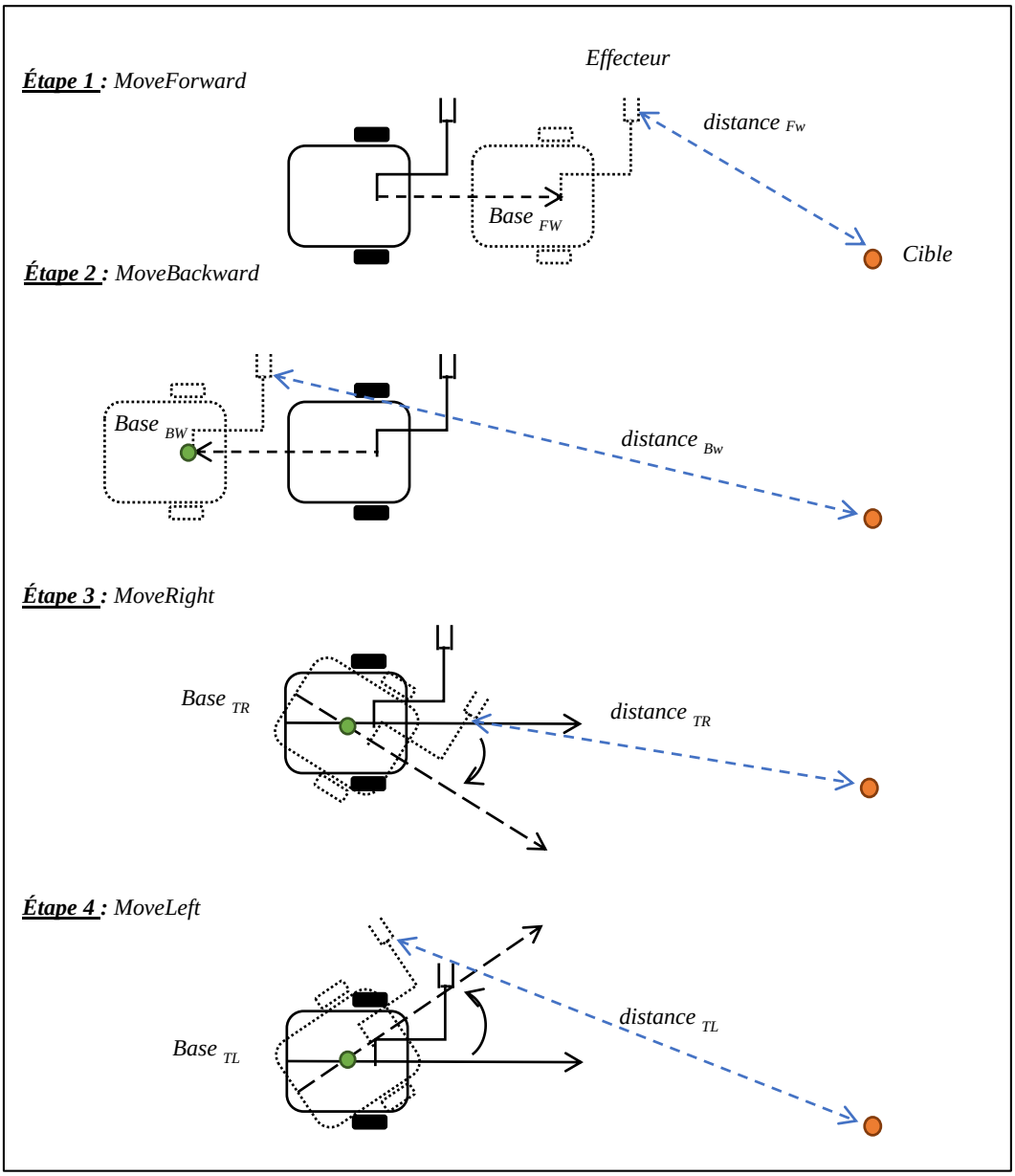


Figure 36. Principe de mouvement virtuel-vérification appliqué pour l'Agent Base mobile.

Les consignes de déplacement issues de l'*Agent Base mobile* sont calculées en utilisant un mécanisme de mouvement virtuelle-vérification, similaire à celui des Agents *Axes*. Cependant, Quatre types de mouvements élémentaires différents sont possibles pour la base mobile *(i) MoveForward* (mouvoir en avant), *(ii) MoveBackward* (mouvoir en arrière), *(iii) TurnRight* (tourner à droite) et enfin *(iv) TurnLeft* (tourner à gauche). Pour la sélection du meilleur déplacement les étapes suivantes sont à considérer :

1. L'*Agent Base mobile* fait un mouvement virtuel en avant *(MoveForward)* avec un pas de déplacement fixe qui est égale à *BaseTranslationFootstep*.
2. Il calcule la nouvelle valeur de la fonction objective, f_{Obj}, entre la nouvelle situation de l'*Effecteur* et celle la *Cible*. Cette valeur dépendra de $Base_{Fw}(x_B, y_B, \theta_B)_{Fw}$, qui est la nouvelle situation de la base mobile après avoir effectué le mouvement virtuel, et *Configuration*$(q_1, \ldots, q_n)$, qui représente la configuration courante du manipulateur.
   - *MoveForward* [*BaseTranslationFootstep*, $Base_{Fw}(x_B, y_B, \theta_B)_{Fw}$, *Configuration* $(q_1, \ldots, q_n)$],
3. L'agent répète, ensuite, successivement ces deux premières actions pour les trois autres mouvements élémentaires. Les deux paramètres qui changent, d'un mouvement virtuel à un autre, sont le pas utilisé ainsi que la direction du mouvement. Ils sont donnés comme suit :
   - *MoveBackward* [*BaseTranslationFootstep*, $Base_{Bw}(x_B, y_B, \theta_B)_{Bw}$, *Configuration* $(q_1, \ldots, q_n)$],
   - *TurnRight* [*BaseRotationFootstep*, $Base_{TR}(x_B, y_B, \theta_B)_{TR}$, *Configuration* $(q_1, \ldots, q_n)$],
   - *TurnLeft* [*BaseRotationFootstep*, $Base_{Tl}(x_B, y_B, \theta_B)_{TL}$, *Configuration* $(q_1, \ldots, q_n)$].

Après avoir évalué les quatre mouvements possibles, l'*Agent Base mobile* sélectionne la nouvelle situation de la base mobile $Base_{Nvlle}(x_B, y_B, \theta_B)_{Nvlle}$. Cette dernière correspond au mouvement qui minimise la valeur de f_{Obj}. Il faut noter que, comme pour le cas de l'*Agent Axe*, le meilleur choix peut être de rester dans la même position actuelle. Ce cas figure se produit lorsque la valeur de f_{Obj}, qui correspond à la situation actuelle, est meilleure que celles des fonctions objectives correspondantes aux quatre mouvements virtuels. À la fin de chaque itération, l'*Agent Base mobile* envoie son *meilleur choix*, $Base_{Nvlle}$ ainsi que la valeur de f_{Obj} à l'*Agent Superviseur*.

Le diagramme d'activité de la figure qui suit, présente un schéma offrant plus d'explication quant à la sélection de la meilleure situation pour la base mobile. De même pour les *Agents Axes*, l'*Agent Base mobile*, connait trois situations. Après sa création, il rentre dans un état d'attente jusqu'à la réception d'un message envoyé par l'*Agent superviseur*. Selon le type de message reçu, l'*agent Base mobile* adaptera son comportement.

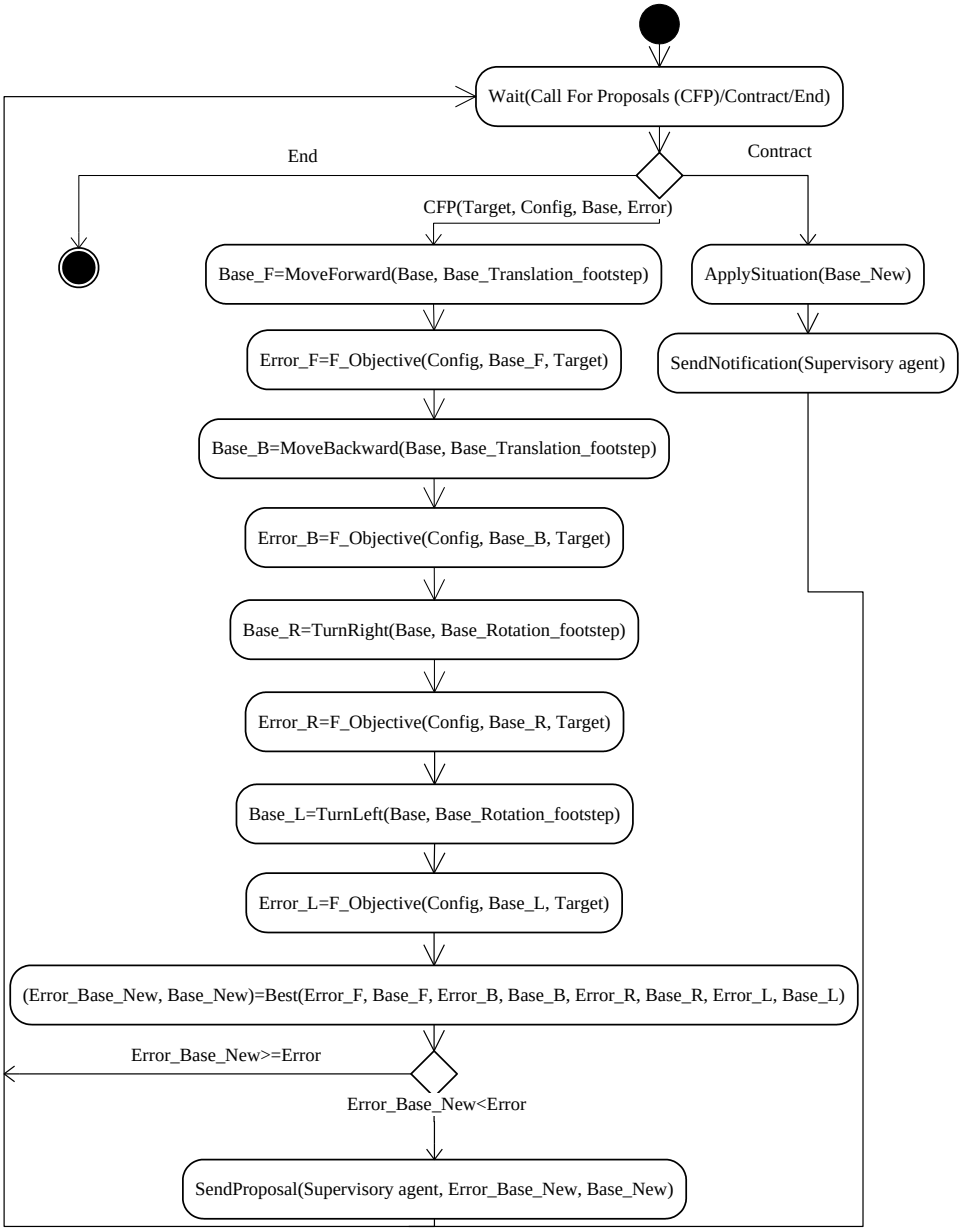


Figure 37. Diagramme d'activité qui détaille le comportement de l'*Agent Base mobile*.

3.3.3.3. **Agent Superviseur**

L'Agent-*Superviseur* possède un comportement hybride. C'est un agent qui réunit des capacités de cognitions, lors de la réalisation des tâches complexes, avec un comportement réactif, pour la sélection de la meilleure proposition parmi celles issues des différents agents de contrôle.

Cet agent est chargé de la synchronisation et de la gestion des données de contrôle échangées entre les différents agents de contrôle et, éventuellement, avec le robot réel à travers la classe *Robot Virtuel*. Il représente, ainsi, le lien entre le système de contrôle et l'opérateur, qui est chargé d'introduire, au système de contrôle, la situation initiale du robot et celle de la cible désirée.

Après la réception des coordonnées de la cible imposée par l'opérateur, l'*Agent Superviseur* vérifie si elle est atteignable, c'est-à-dire si les coordonnées de la cible font partie de l'espace de travail atteignable du robot. Si ce n'est pas le cas (la cible n'est pas atteignable), il arrête le processus de contrôle. Dans le cas normal, l'*Agent Superviseur* calcule la valeur de la fonction objective f_{Obj} qui correspond à la distance initiale *(*$distance_{Init}$*)* entre l'effecteur du robot et les coordonnées de la *Cible*. Ensuite, il transmit cette valeur (*distance*$_{Init}$) à tous les agents de contrôle *(Agent Base mobile, Agent Axe*$_1$*, … , Agent Axe*$_n$*)* conjointement avec la situation initiale de la base mobile (*Base*$_{Init}$) et la configuration initiale des articulations (*Configuration*$_{Init}$*).* Ces informations constituent un message de type *Call For Proposals*. Après, l'*Agent Superviseur* se met dans un état d'attente des propositions (*proposals*) issues des agents de contrôle.

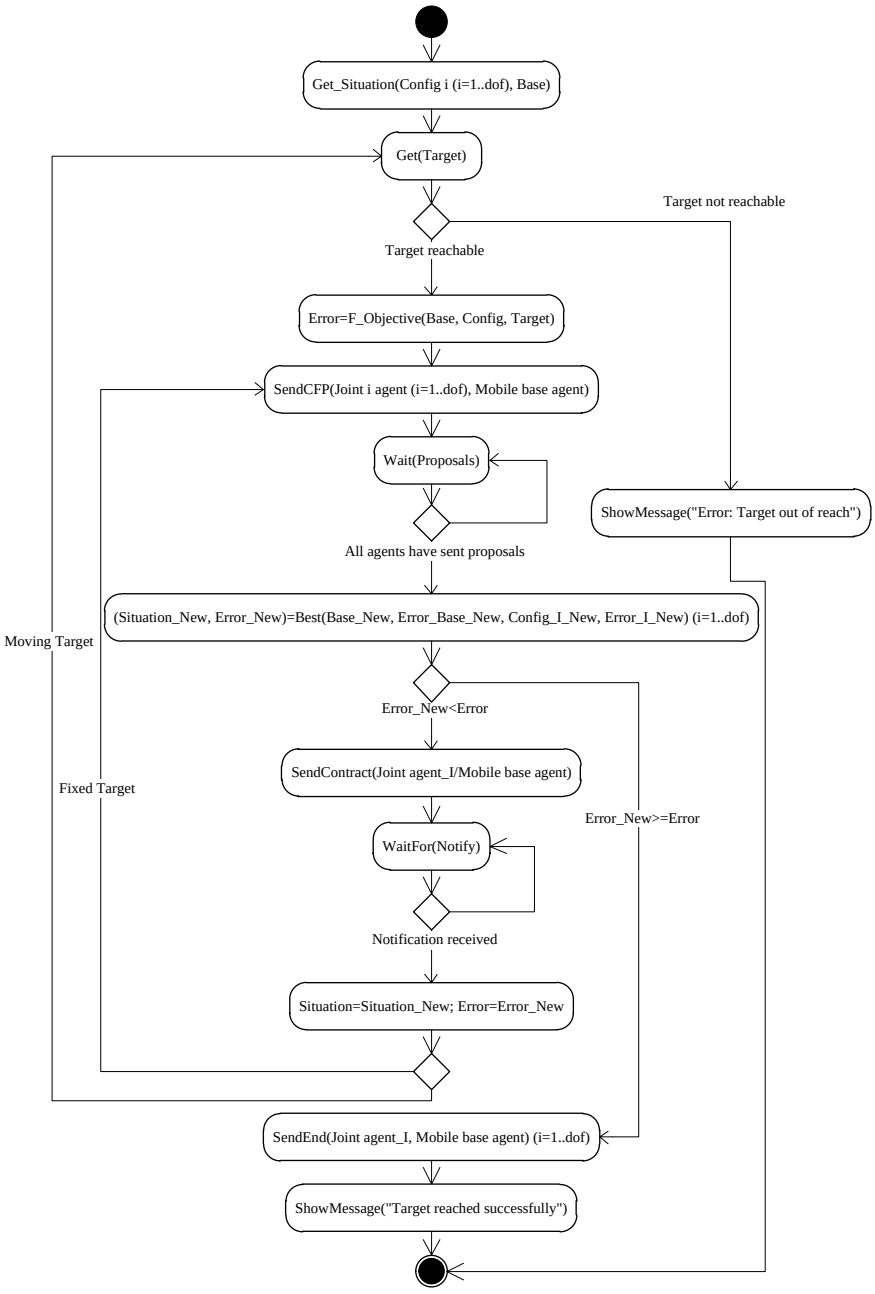


Figure 38. Diagramme d'activité de l'*Agent Superviseur*.

Une fois que les données requises soient réunies, les agents de contrôle (*Agent Base mobile* et les *Agents Axes*) vont effectuer leurs traitements respectifs dans le but de choisir la meilleure nouvelle configuration du robot. Ces traitements sont exécutés en parallèle et d'une manière complètement indépendante d'un agent à un entre ce qui va améliorer considérablement les performances et augmenter l'efficacité.

Par la suite, l'*Agent Superviseur* reçoit les réponses des agents de contrôle contenant les meilleures propositions locales de chaque agent. Ces réponses représentent les meilleures propositions pour la configuration des articulations du manipulateur, issues des *Agents Axes*, ainsi que la meilleure proposition pour la situation de la base mobile issue de l'*Agent Base mobile*. L'*agent Superviseur* sélection, ensuite, la meilleure proposition qui minimise la valeur de f_{Obj} ($distance_{Nvlle}$). Il envoie, par la suite, un message de type *Contract (Accept Proposal)* à l'agent correspondant pour exécution. Si cette valeur de $distance_{Nvlle}$ correspond à la valeur optimale, l'*Agent Superviseur* interromprait le processus de contrôle en envoyant des messages *End Task* aux autres agents. Autrement, il réitère le processus de contrôle en envoyant des nouveaux messages *Call For Proposals* aux autres agents de contrôle.

3.3.4. Synthèse

Nous avons présenté dans cette partie une méthode basée sur un SMA constitué d'un ensemble d'agents réactifs (les *Agents Axes*) et de deux agents hybrides (l'*Agent Base mobile* et l'*Agent Superviseur*), pour le contrôle d'un manipulateur mobile. Dans un tel système, tous les agents travaillent en parallèle et d'une manière indépendante. L'*agent Superviseur* est responsable d'assurer la synchronisation entre ces différents agents. Le schéma de contrôle globale, ainsi que les interactions entre les différentes entités du système, est illustrée dans le diagramme de séquence dans la figure qui suivant.

Le système tel qu'il est conçu n'est pas sensible aux différentes pannes qui peuvent surgir. Ceci s'explique par le fait que les agents de contrôle soient totalement indépendants et qu'aucun échange direct de messages n'est prévu entre ces agents. La synchronisation est assurée par l'*Agent Superviseur*. Si un agent de contrôle ne répond pas, les autres agents essayent indépendamment de recouvrir cette anomalie et satisfaire, ainsi, l'objectif désiré.

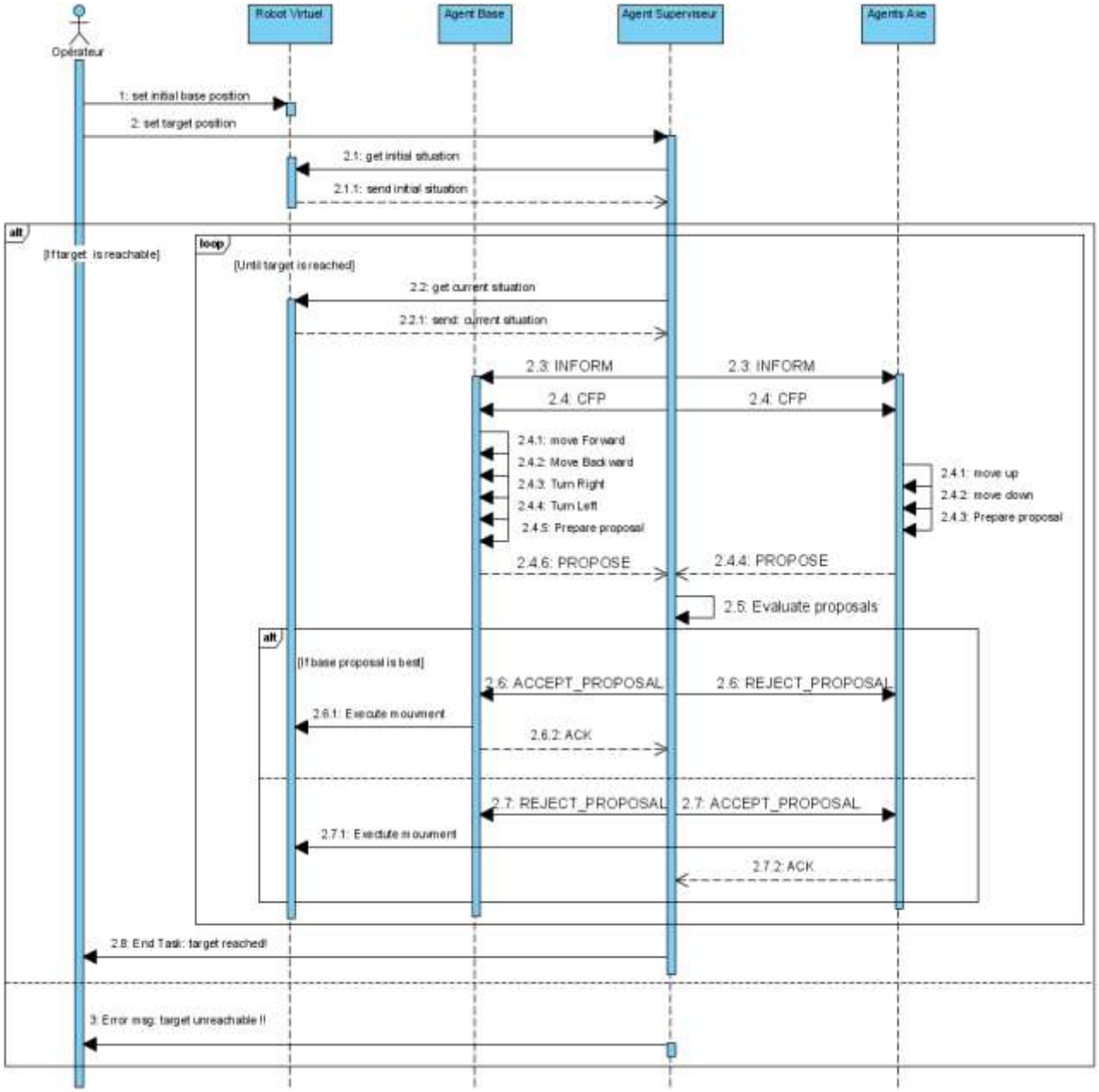


Figure 39. Diagramme de séquence globale

3.4. Implémentation du système

3.4.1. Outils d'implémentation

Pour l'implémentation de notre approche de contrôle, nous avons choisi le langage de programmation orienté objet Java (Java2 Standard Edition). Cela nous a offert un environnement de développement complet et bien adapté à la programmation des applications de bureau (Desktop Applications). Ce choix est justifié par (*i*) la panoplie de fonctionnalités proposées avec les bibliothèques fournies par le JDK (Java Development Kit) ainsi que celles développées par des communautés internationales (*ii*) le simulateur graphique mis à notre disposition est développé lui-

même avec le langage Java (voir annexe). Nous avons utilisé l'IDE *Netbeans* pour la mise en œuvre expérimentale de notre approche.

Nous nous somme basés sur la plateforme open source *jade* (***J**ava **A**gent **DE**velopment framework*) **[**33**]** [32] comme outil pour l'implémentation du système multi-agents. Cet outil est considéré comme un middleware qui offre une couche intermédiaire pour simplifier la réalisation d'applications distribuées qui se basent sur le concept d'agent [22]. La plateforme *jade* présente un environnement de création et de manipulation des agents ainsi qu'un ensemble d'outils graphiques pour le monitoring et le débogage d'applications.

3.4.2. Interface de simulation

Pour des besoins de validation de l'approche, nous avons développé une interface graphique pour la simulation des scénarios. Cette interface, présentée dans la figure ci-dessous, nous permis de lancer la simulation d'un ou plusieurs scénarios à la fois. Un scénario de validation est décrit par les informations suivantes :

- Le numéro de scénario (Scenario Number).
- La position de la cible (Target Position) donnée par (X_{EFinal}, Y_{EFinal}, Z_{EFinal}).
- La situation initiale de la base mobile (Mobile base initiale position) donnée par (X_{B_init}, Y_{B_init}, $Theta_{B\_init}$).
- La configuration initiale du manipulateur (Manipulator Initial Configuration) donnée par (Q_{1_init}, …, Q_{6_init}).

Dans cette approche, nous considérons que les pas (les *footsteps*) sont fixes. Toutefois, des mécanismes permettant une sélection dynamique des pas ou un ajustement au cours de l'exécution peuvent être réfléchis. De ce fait, une étude détaillée dans la suite de ce chapitre explique le choix statique pour les différents pas possibles. Comme nous l'avons défini dans la partie conception, chaque agent de contrôle de type *Agent axe* possède un seul pas *JointsFootstep*, alors que l'agent de contrôle *Agent Base* possède deux pas différents (*i*) *Base Translation Footstep* et (*ii*) *Base Rotation Footstep*.

Pour lancer la simulation d'un scénario via l'interface graphique, la liste des informations citées précédemment doit être introduite avant de cliquer sur le bouton *Start*. Plusieurs instances, pour des scénarios différents, peuvent être lancées simultanément. La partie inférieure de l'interface graphique est une console pour l'affichage de messages liés au déroulement et à l'exécution des différentes tâches.

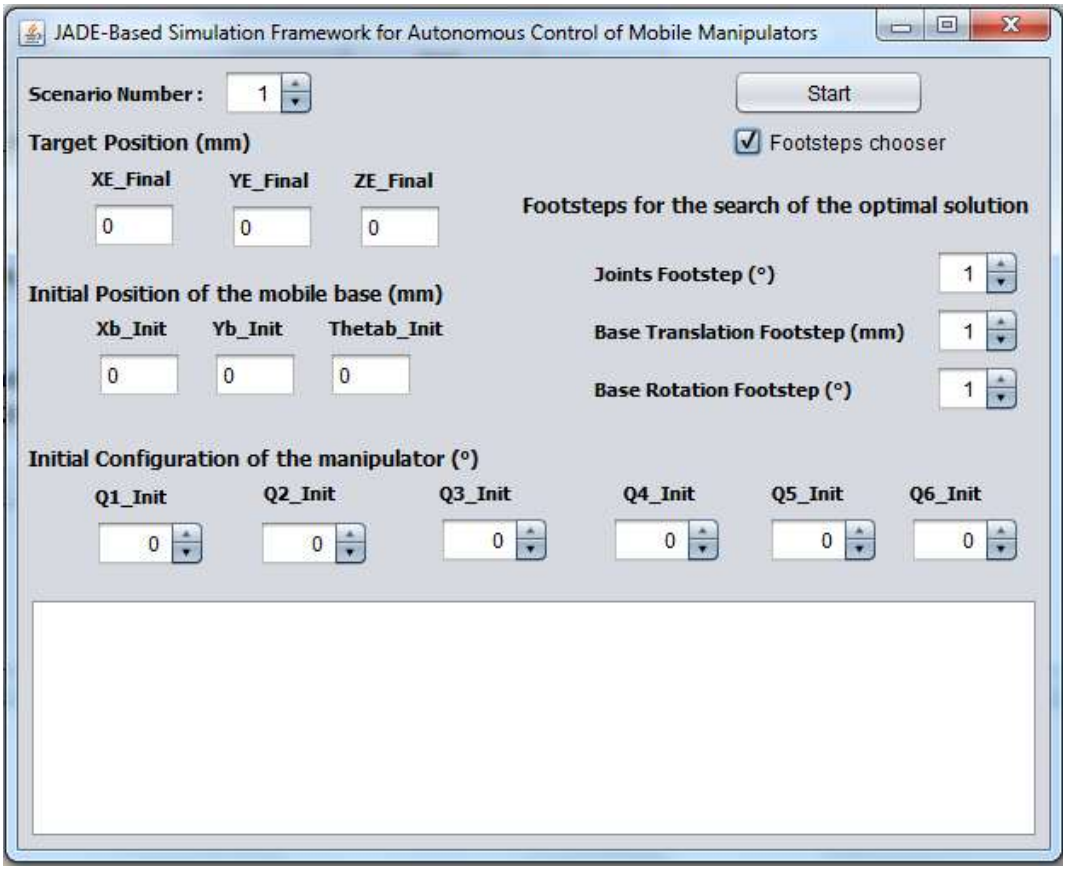


Figure 40. Interface graphique principale pour la plateforme de simulation.

La figure suivante montre l'interface graphique, offert par *jade*, pour la gestion d'une instance d'un scénario donné. Nous pouvons distinguer les agents système utilisés par la plateforme *jade* ainsi que ceux de l'approche (*Agent Superviseur*, *Agent Base mobile* et les six *Agents Axes*). Pour chaque instance du programme lancée, une interface similaire est affichée en concordance.

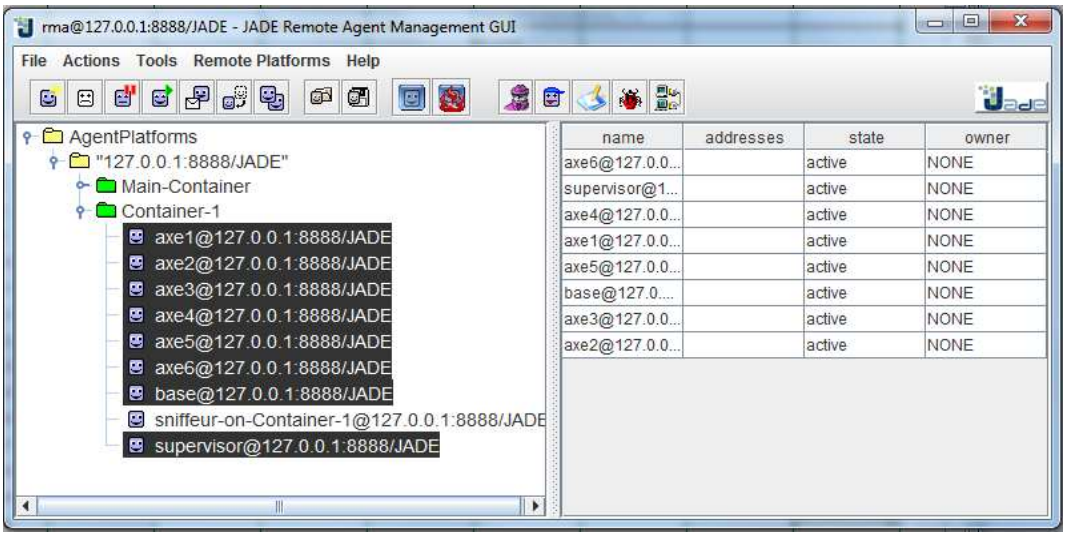


Figure 41. Interface jade *Remote Agent Management GUI* affichée suite au lancement d'une instance

3.4.3. Échange de messages

Les agents *jade* sont réunis pour assurer les tâches qui leurs étaient affectées. Chaque agent est censé réaliser un traitement propre à lui et d'une manière totalement indépendante. Cela n'est garanti qu'avec une exécution autonome pour chaque agent. Par conséquent, un comportement global, permettant la réalisation des tâches complexes, émerge de l'interaction entre les différents agents. Dans *jade*, cette interaction est réalisée via un modèle de communication basé sur un échange de message. Ces messages ont un format précis, spécifié par le langage *ACL* (*Agent Communication Language*) défini par FIPA (Foundation for Intelligence Physical Agents) [20] et qui représente un langage de communication entre agents. Les principales données circulées dans chaque message de communication sont :

- L'émetteur ainsi que le récepteur du message.
- Le type de message, qui représente aussi sa nature et son objectif.
- Le contenu du message, qui représente l'information circulée à travers ce message.

Les types de message les plus usuels sont INFORM, REQUEST, PROPOSE, etc. Une combinaison de cette liste nous permettra de spécifier une(des) séquence(s) prédéfinie(s) de messages échangés entre les agents lors d'une communication. Chaque type de message sert à un objectif bien particulier. Un message de type INFORM est utilisé pour informer le récepteur d'un fait ou une action. Le message de type REQUEST est envoyé lorsque l'émetteur veut que le récepteur exécute une action donnée. Le type QUERY a comme but de récupérer une donnée ou de recevoir une information particulière.

La plateforme *jade* offre aussi la possibilité d'utiliser, lors de l'implémentation d'un *SMA*, des protocoles qui régissent l'échange de messages entre les agents évoluant dans le même système ou bien qui appartient à des systèmes distincts. Pour notre cas de figure, nous nous sommes basés dans notre implémentation sur le CNP (*Contract Net Protocol)* comme protocole d'allocation de tâches entre l'agent superviseur et les agents de contrôle. Le protocole CNP est basé sur une structure décentralisée où les agents peuvent avoir deux rôles distincts (*i*) *Manager*, ou (*ii*) *Contractor*. Ce protocole est utilisé comme mécanisme d'allocation et de routage dynamique des tâches dans les approches qui se basent sur le paradigme agent. Le protocole CNP suit les étapes suivantes :

- L'agent *Manager* annonce un call for proposal (*CFP*).
- Les agents *Contractors* soumettent leurs propositions en réponse à l'annonce de l'agent *Manager*.
- L'agent *Manager* évalue les offres soumises, puis procède à l'attribution du contrat proposé à l'agent *Contractor* avec l'offre le plus adéquat.

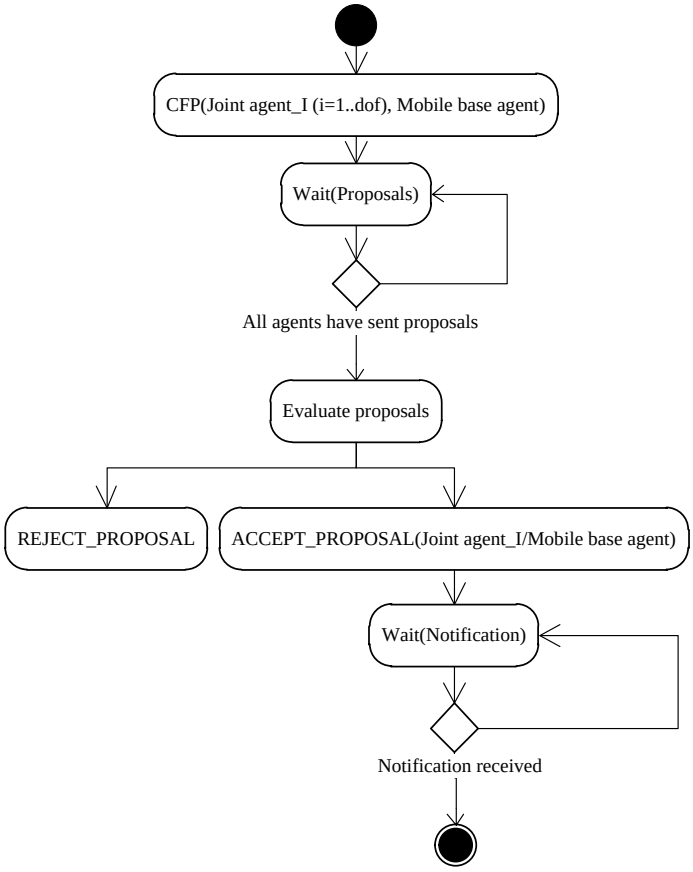


Figure 42. Comportement dynamique pour l'*Agent superviseur* basé sur un *Contract-Net*

L'acheminement des étapes décrites dans ce qui précèdent est projeté dans le diagramme de la figure précédente, où nous présentons un *Contract-Net* qui reflète le déroulement assuré par l'agent superviseur. Ce dernier représente l'agent *Manager*, tandis que les agents contrôles sont des agents *Contractors*. Le processus que nous décrivons se déroule en respectant les étapes décrites ci-dessous :

1. Annonce du contrat, qui est définie par un CFP (Call For Proposal) destiné aux agents de contrôle (agent base mobile et agents axes).
2. Attente des réponses (les propositions), où chaque agent de contrôle est censé annoncer sa proposition concernant le CFP. L'agent superviseur va attendre, pour une période déterminée, la réception de toutes les propositions faites par les agents de contrôle.
3. Évaluation des propositions, une fois les annonces reçus, et sélection de la meilleure proposition selon un critère prédéfini.
4. Notification et attribution du contrat à l'agent de contrôle avec la meilleure proposition.

Pour implémenter les différentes interactions entre les agents de contrôle (agent base mobile et agents axes) et l'agent superviseur, nous avons proposé une définition de plusieurs type de

messages. Cette définition est basée sur les messages de base offerts par *33*. Les principaux types de messages que nous avons utilisés sont les suivants :

- *INFORM* : au début de chaque itération, l'agent superviseur envoie un message de ce type vers tous les agents de contrôle pour les informer de la nouvelle situation du système. Les données véhiculées à travers un message de ce type sont (*i*) la situation courante de la base mobile (x_B, y_B, θ_B) ainsi que (*ii*) la configuration actuelle du manipulateur $(q_1, \ldots, q_n)$.
- *CFP (Call For Proposal)* : ce type de messages a pour objectif de solliciter des nouvelles propositions pour le prochain mouvement à entreprendre par le robot. L'agent superviseur, via un message de ce type, diffuse vers les agents de contrôle actifs la situation de la cible imposée (x_C, y_C, z_C) lorsque la position de la cible change. Ce message est envoyé une seule fois lorsque la tâche à accomplir considère une cible statique. Cependant, dans le cas de suivi d'une cible en mouvement, le message CFP est envoyé à chaque itération vers les agents de contrôle, juste après l'envoie du message INFORM.
- *PROPOSE* : ce message est envoyé par les agents de contrôle à l'agent superviseur. Il représente une réponse au message CFP envoyé par ce dernier agent. Ce type de message est envoyé après avoir choisi la meilleure proposition locale. Il comporte alors une proposition de la meilleure valeur, choisie localement par chaque agent de contrôle. La sélection de la valeur envoyée est faite en se basant sur les données reçues par les agents de contrôle à travers les deux messages INFORM et CFP.
- *ACCEPT_PROPOSAL* : suite à la récolte des propositions faites par les agents de contrôle, ce message est envoyé après la sélection de la meilleure proposition par l'agent superviseur. Ce dernier est chargé d'envoyer ce message à l'agent qui a soumis la meilleure proposition pour lui confirmer l'acceptation de cette dernière et éventuellement son exécution.
- *REJECT_PROPOSAL* : un message d'infirmation est envoyé par l'agent superviseur vers les agents de contrôles dont la proposition qui l'ont transmise, via le message PROPOSE, n'était pas sélectionnée. Pour chaque message ACCEPT_PROPOSAL envoyé, (n-1) messages de REJECT_PROPOSAL sont transmis, tel que *n* est le nombre des agents de contrôle actuellement actifs dans le système.

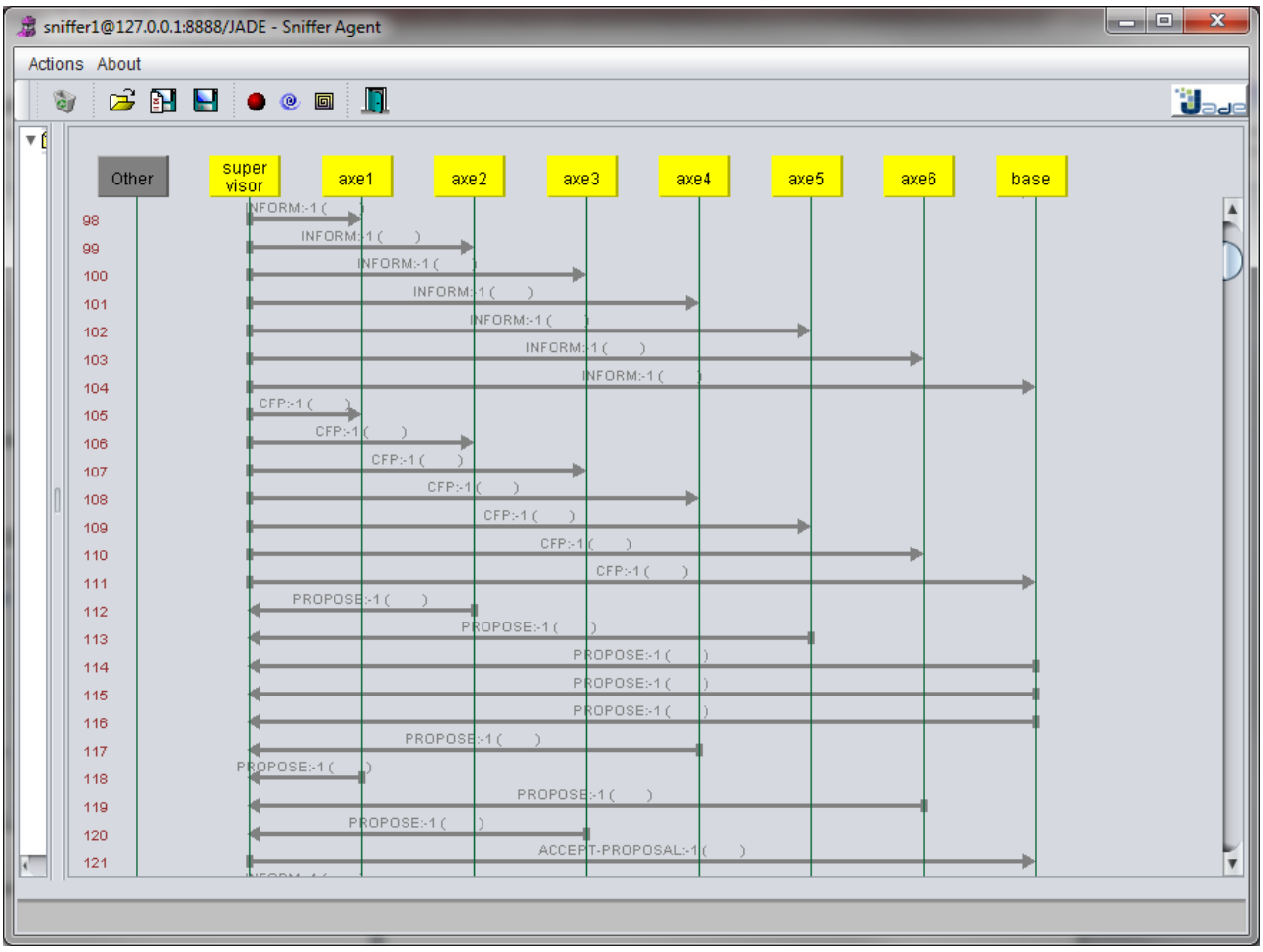


Figure 43. Interface jade-Sniffer montrant un échange de messages entre les agents du système

3.5. Conclusion

Ce chapitre a commencé par la description des spécifications globales du système robotique expérimental, en présentant sa structure matérielle et les deux sous-systèmes hétérogènes qui le composent (base mobile et bras manipulateur). Ensuite, nous avons fait une analyse détaillée des techniques et concepts utilisés pour le déploiement d'un nouveau système de contrôle basé sur une architecture multi-agents. Cette première partie est conclue par une analyse fonctionnelle de l'approche adoptée.

Dans une deuxième section et après avoir exposé les différentes vues de l'approche, ainsi que sa composition logique, nous avons présenté les différents schémas de contrôle qui sont envisageables pour le contrôle des manipulateurs mobiles. L'analyse conceptuelle a été effectuée en détaillant le principe de *mouvement virtuel-vérification* utilisé dans l'implémentation des agents de contrôle. Ce même principe se diffère d'un agent de contrôle à un autre. De ce fait, il est implémenté de deux différentes manières, selon le comportement de l'*Agent Base mobile* et celui des *Agents Axes*. L'*Agent Superviseur* assure, pour chaque itération, la sélection du meilleur mouvement à entreprendre.

En souhaitant que la conception de l'architecture proposée soit la plus générique possible, nous avons présenté un schéma de contrôle qui soit facilement adaptable au contrôle de n'importe qu'elle structure robotisée. Cette intention, de concevoir un tel schéma, nous a incité à exploiter le paradigme multi-agents qui a prouvé être un concept très intéressent lors de la conception des architectures de contrôle sans avoir recours à l'étude des modèles mathématiques complexes.

Dans ce chapitre, nous avons aussi présenté les détails liés à l'implémentation et la mise en œuvre de la nouvelle approche de contrôle. Cette dernière offre une alternative aux approches roboticiennes classiques qui se basent sur l'étude et le calcul des modèles géométrique des robots. Dans l'approche proposée, le système robotique est considéré dans sa totalité, sans fixer des limites conceptuelles entre les deux structures mécaniques hétérogènes qui le composent (en l'occurrence la base mobile et le manipulateur).

Nous nous sommes basés, pour la concrétisation de l'approche, sur le paradigme agent, où chaque sous-système est contrôlé par un agent. Chaque agent du système opère indépendamment en réponse à des changements dans son environnement. Un comportement global émergera de l'exécution parallèle des agents, ce qui permet au robot d'accomplir des tâches complexes.

Le présent chapitre décrit, dans une dernière partie, les outils et les plateformes de développement exploitées ainsi que l'interface de simulation utilisée. Enfin, dans cette même partie, nous avons défini un protocole de communication à travers un échange de différents types de messages.

Dans ce qui suit, nous présentons les tests et les expérimentations afin de valider notre approche. Nous montrons aussi les résultats des différents scénarios de simulation, ainsi qu'une étude comparative des résultats obtenus avec d'autres travaux de la littérature.

CHAPITRE 4
EXPÉRIMENTATIONS ET VALIDATION

4.1. Introduction

Dans le chapitre précédent, nous avons présenté une étude pratique afin de mettre en œuvre une approche de contrôle basée sur un système multi-agents. Nous avons montré les différents vues et spécifications, pour un tel système, permettant de générer les consignes de contrôle pour un manipulateur mobile dans l'accomplissement des tâches qui lui sont affectées.

Le présent chapitre décrit les différents scénarii de validation et le choix de ces derniers. Les résultats de simulations, réunies dans des plusieurs tableaux récapitulatifs, seront présentés via des histogrammes et des schémas illustratifs. Pour situer notre travail par rapport à l'état de l'art, les résultats obtenus seront exposés et comparés avec celles de la littérature.

4.2. Scénarios de validation

Pour les besoins de validation de l'approche proposée, nous avons choisi différents scénarios. Toutefois, le choix de ces scénarios, dans la globalité, n'était pas arbitraire. Chaque scénario choisi sert à un objectif de test bien particulier et répond, en conséquence, à un besoin de validation. Chacun de ces scénarios est défini par les paramètres de simulation suivants :

- La situation initiale de la base mobile $Base_{Init}(X_B, Y_B, theta_B)_{Init}$.
- La configuration initiale du manipulateur $Configuration_{Init}(q_1, q_2, q_3, q_4, q_5, q_6)_{Init}$.

- La position de la cible à atteindre $Effector_{Fin}(x_E, y_E, z_E)_{Fin}$.

Les performances de l'approche sont jugées à travers les résultats obtenus de la simulation. Selon la combinaison des pas choisis, et pour chaque scénario donné, les informations que nous pouvons récoltées et évaluées, à la fin de chaque instance de simulation, sont données comme suit :

- La valeur de l'erreur minimale de la distance entre la position de l'effecteur et celle de la cible imposée.
- Le nombre total d'itérations.
- La situation finale de la base mobile et la configuration finale des articulations du manipulateur.

Le tableau suivant regroupe les cinq scénarios que nous avons choisis pour la validation de l'approche de contrôle proposée. Chaque scénario est considéré pour tester une situation particulière ou bien pour valider un comportement spécifique. Toutefois, il faut noter que le choix de ce tableau n'est pas arbitraire, de fait que ces mêmes scénarios ont été invoqués dans la littérature. Ceci est dans le but d'offrir un support de comparaison avec les résultats de notre approche dans la partie Étude comparative.

Scénarios	$Base_{Init}(X_B, Y_B, theta_B)_{Init}$	$Effector_{Fin}(x_E, y_E, z_E)_{Fin}$ (mm)	Fct Obj $_{Init}$ (distance) (mm)
1	(0, 0, 0, 0, 0, 0)	(-330, -630, 1080)	1126.9129
2	(0, 0, 0, 0, 0, 0)	(-4260, 0, 665)	4698.9355
3	(0, 60, 0, 0, 32, 0)	(-2408, -108, 1472)	3114.8048
4	(0, 87, 0, 0, 5, 0)	(-2400, -63, 1325)	2946.8779
5	(0, 87, 0, 0, 5, 0)	(-2400, -67, 1320)	2946.8226

Tableau 1. Paramètres initiaux de simulation pour les cinq scénarios choisis.

Les deux premiers scénarios ont la même configuration initiale du manipulateur et aussi la même situation initiale de la base mobile. Néanmoins, l'objectif à atteindre dans le premier scénario fait partie de l'espace de travail actuel du manipulateur. Pour le deuxième scénario, l'objectif à atteindre est en dehors de l'espace courant du robot. Dans ce cas, il faut faire mouvoir la base mobile à une nouvelle situation afin de pouvoir positionner l'effecteur du robot dans cette position imposée. La distance initiale entre la position de l'effecteur et celle de la cible imposée dans le deuxième scénario est plus grande (1126mm dans le premier scénario contre 4698mm pour le

deuxième). En ce qui concerne le troisième scénario, la distance initiale vers la cible est importante, mais la configuration du manipulateur est différente par rapport au deuxième scénario. Donc, nous avons choisi les trois premiers scénarios pour étudier l'apport de la situation initiale du robot ainsi que la distance vers la cible imposée.

Les scénarios quatre et cinq ont pour but de valider la précision de notre contrôleur face à des positions cibles qui se rapprochent beaucoup. Dans ces deux scénarios, le robot démarre sa tâche, à partir de la même situation pour la base mobile et avec la même configuration pour le manipulateur, vers deux cibles distinctes qui se trouvent à la même distance mais avec des positions très proches. L'objectif est d'étudier le comportement de notre contrôleur et le chemin suivi pour rejoindre deux cibles imposées très proche l'une de l'autre.

4.3. Expérimentations

Notre approche est basée sur une sélection statique de la combinaison des pas (footstep). Chaque combinaison se compose des trois pas suivants :

- *joint footstep* : représente la rotation élémentaire (en degrés) de chaque articulation.
- *base translation footstep :* représente un déplacement élémentaire (en millimètres) en translation pour l'agent base mobile.
- *base rotation footstep* : représente une rotation élémentaire (en degrés) par rapport la position courante de la base mobile.

Nous avons testé plusieurs valeurs pour des combinaisons multiples des pas afin de trouver la composition optimale. Après plusieurs tentatives, nous avons fixé un intervalle entre 1 et 10, ce qui nous donne plusieurs centaines de possibilités. Puisque une et une seule combinaison n'est permise, il faut chercher la bonne. Donc pour recouvrir le maximum de valeurs possibles dans l'intervalle précédant, nous avons décidé d'opter pour l'ensemble dénombrable $\{1, 5, 10\}$. Les combinaisons des valeurs pour les pas possibles sont au nombre de 27 combinaisons (3x3x3). Il nous reste alors de trouver la meilleure combinaison qui offre la combinaison optimale des pas selon des critères bien définis et pour des applications ciblées.

La meilleure combinaison de pas doit garantir un comportement optimal pour les différents scénarios imaginables, ainsi que pour les évènements inattendus. Ce dernier cas de figure est représenté par la présence de pannes au niveau d'une ou plusieurs articulations du manipulateur ou bien au niveau de la base mobile.

Pour suffisamment approfondir nos résultats et afin de choisir les meilleurs pas, nous avons exécuté 27 instances, pour le même scénario, avec les différentes combinaisons de pas possibles. Ensuite, pour chacune de ces 27 différentes instances, nous avons évalué les deux paramètres de

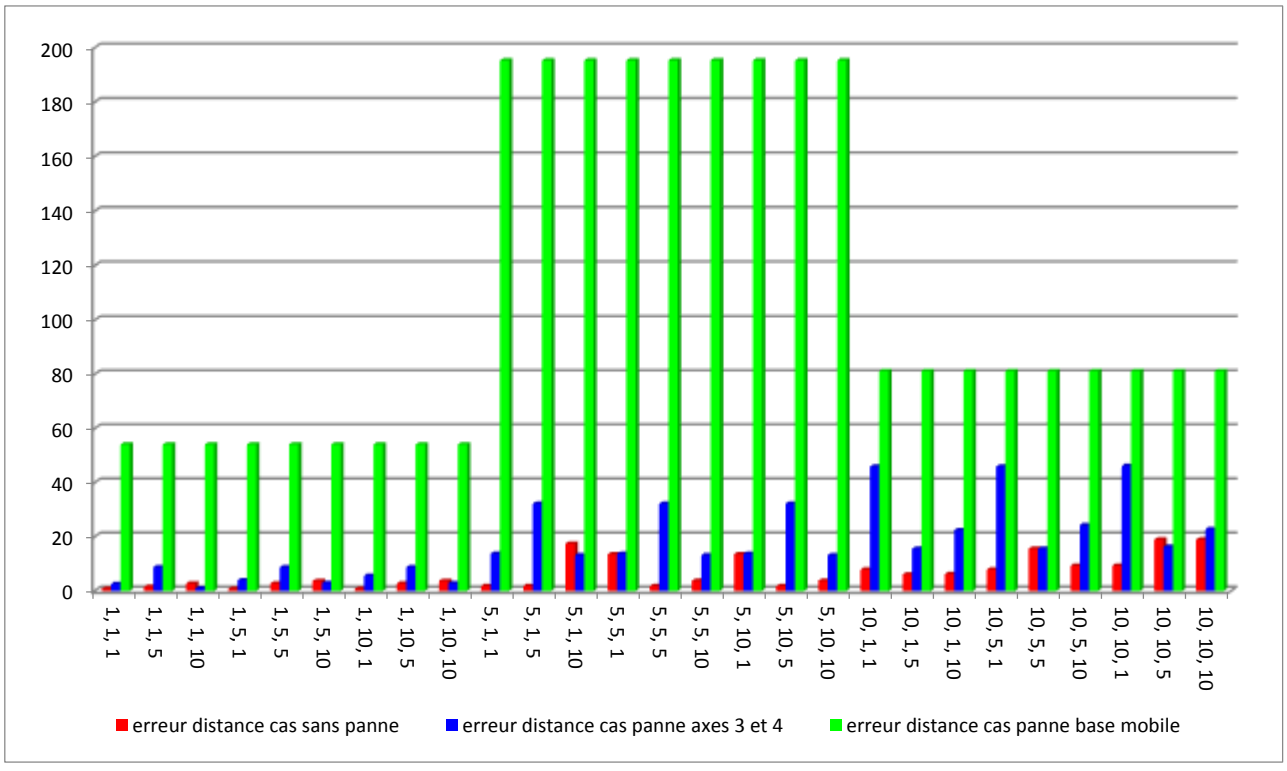
sorties (*i*) *Erreur minimale*, et (*ii*) *Nombre d'itérations*. Nous avons répété ces mêmes tests pour un deuxième cas, où nous simulons la survenance d'une panne au niveau des articulations 3 et 4. Nous avons aussi considéré un troisième cas, où une panne est simulée au niveau de la base mobile.

La procédure que nous venons de décrire est réitérée pour les cinq scénarios présentés dans ce qui précèdent. En se basant sur toutes les informations récoltées, nous avons pu dresser plusieurs histogrammes qui seront présenté dans ce qui suit.

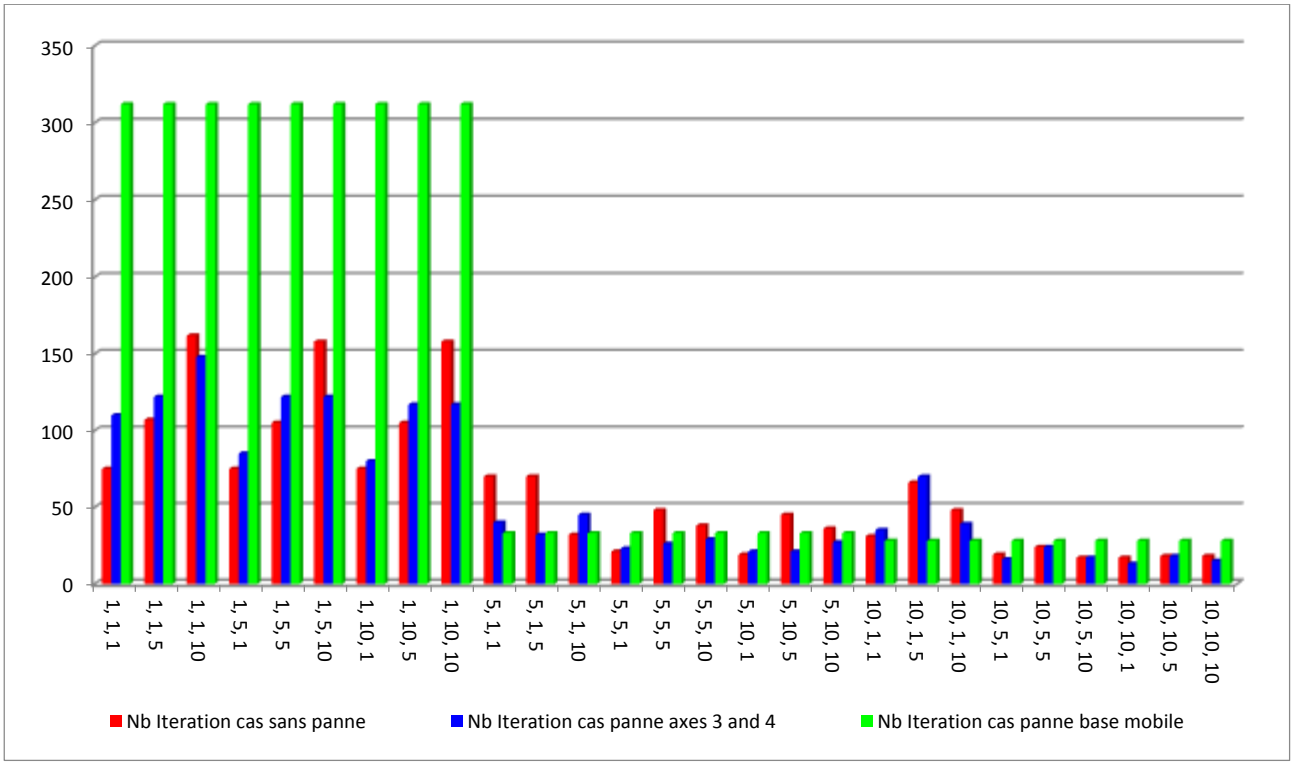
4.3.1. Résultats obtenus

Pour chacun des cinq scénarios, nous avons évalué la valeur de l'erreur en distance entre l'effecteur et la cible imposée ainsi que le nombre d'itérations pour chaque combinaison des pas. Les combinaisons des pas sont de la forme suivante {join footstep, base translation footstep, base rotation footstep}. La liste des 27 triplets des pas possibles est {(1, 1, 1), (1, 1, 5), (1, 1, 10), (1, 5, 1), …, (10, 10, 5), (10, 10, 10)}.

Les deux figures, présentées pour l'évaluation de chaque scénario, représentent respectivement les valeurs finales de l'erreur en distance et le nombre d'itérations. Ceci est évalué pour les 27 combinaisons possibles des pas et pour les trois cas de figures traitées (*i*) sans pannes, (*ii*) en présence d'une panne au niveau des articulations 3 et 4, ainsi que pour (*iii*) un troisième cas, qui simule la panne de la base mobile.

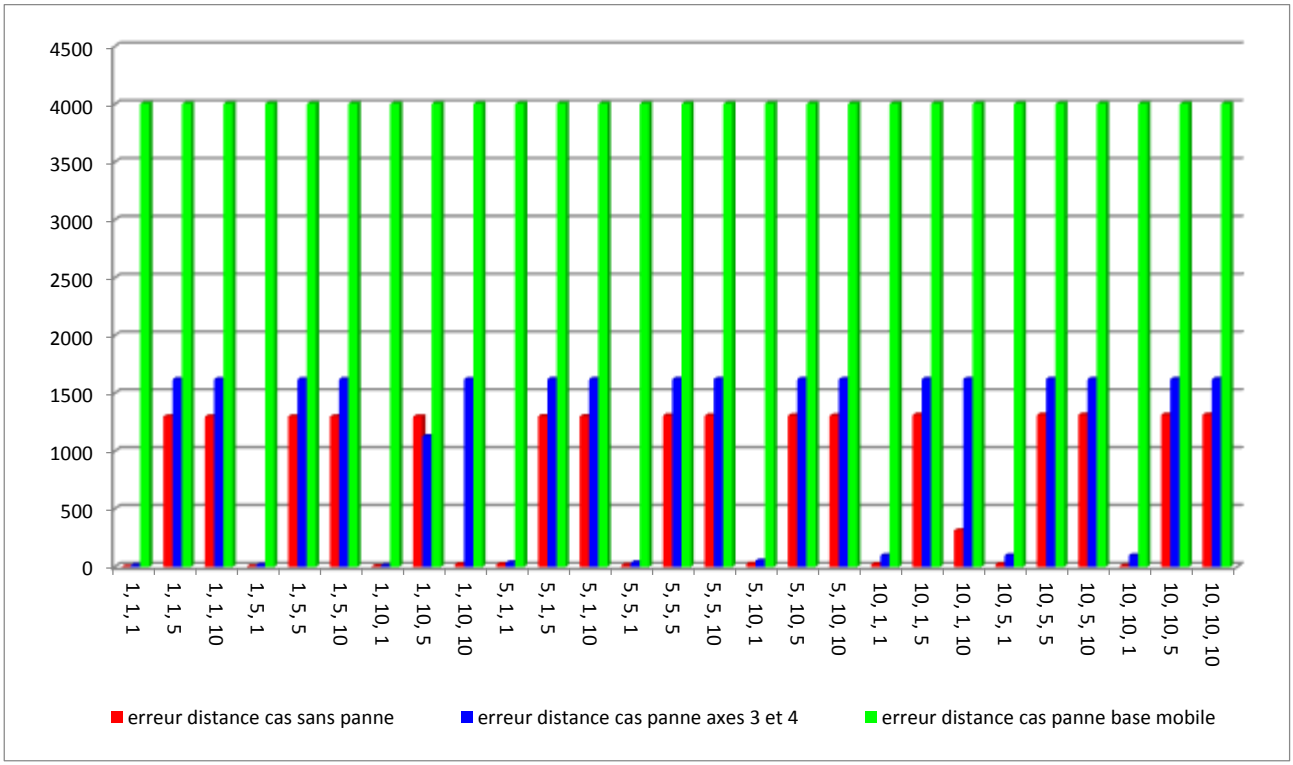


(a) Erreurs en distance pour les 27 combinaisons de pas avec les trois cas de figures

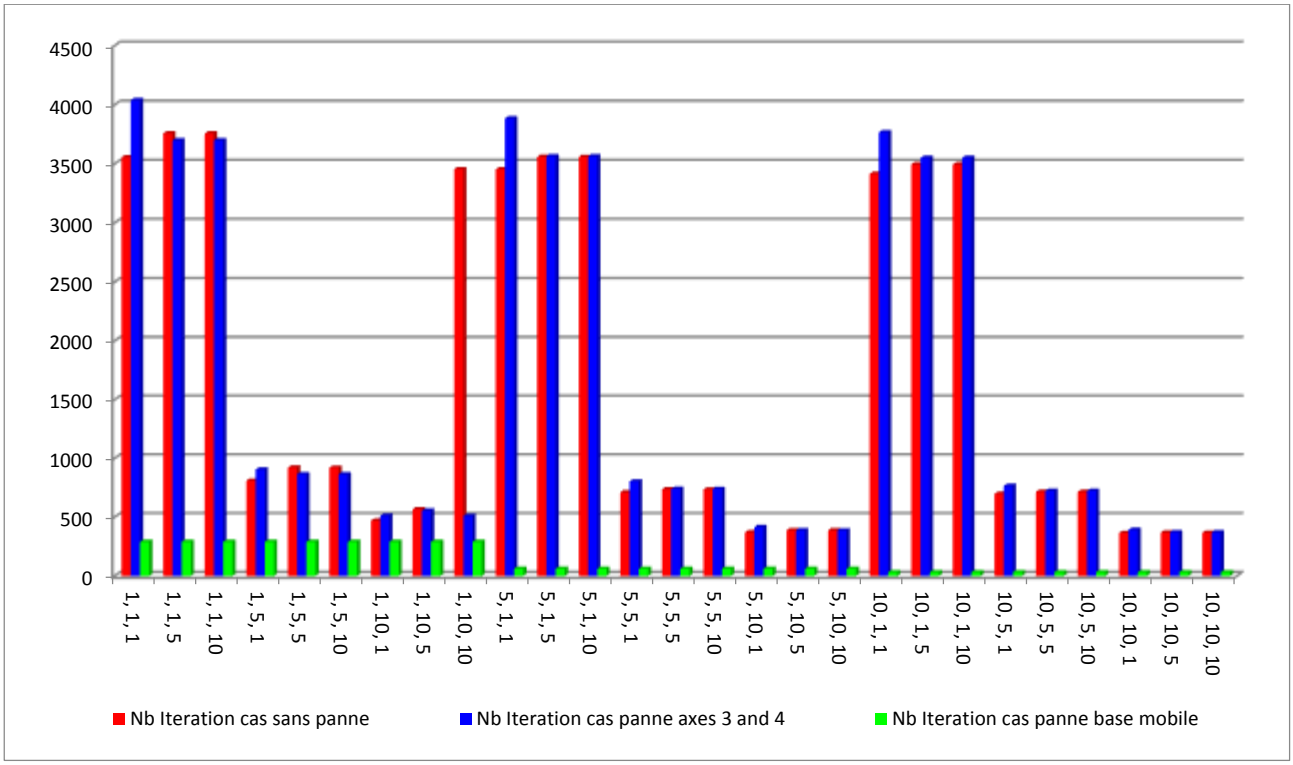


(b) Nombre d'itérations pour les 27 combinaisons de pas avec les trois cas de figures

Figure 44. Résultats obtenus pour le premier scénario

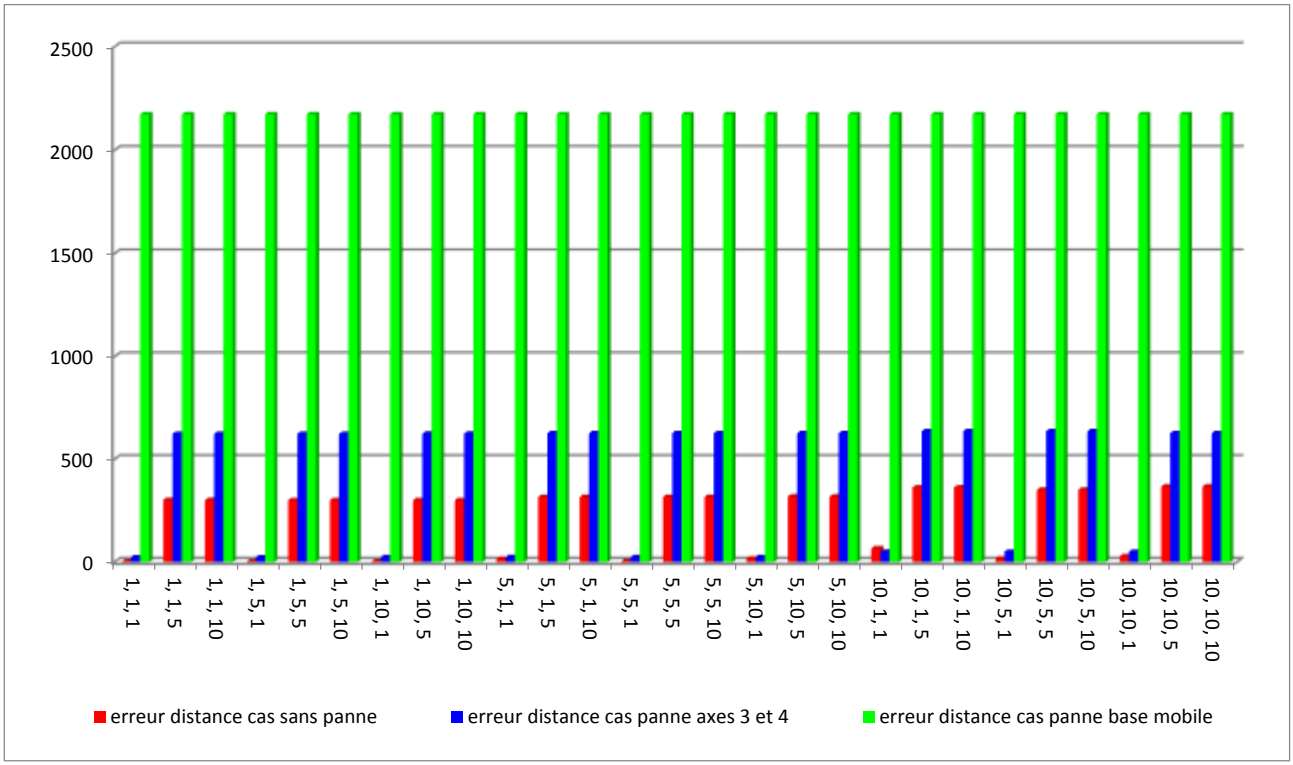


(a) Erreurs en distance pour les 27 combinaisons de pas avec les trois cas de figures

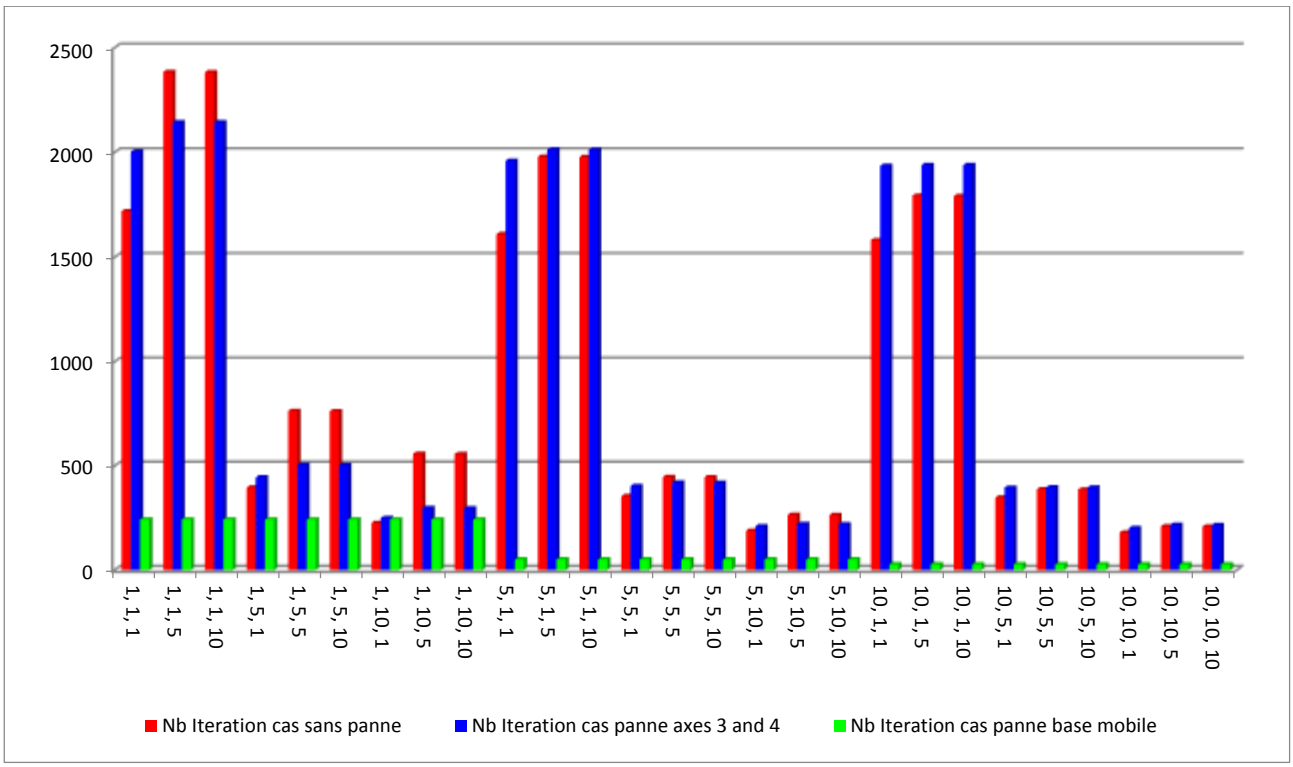


(b) Nombre d'itérations pour les 27 combinaisons de pas avec les trois cas de figures

Figure 45. Résultats obtenus pour le deuxième scénario

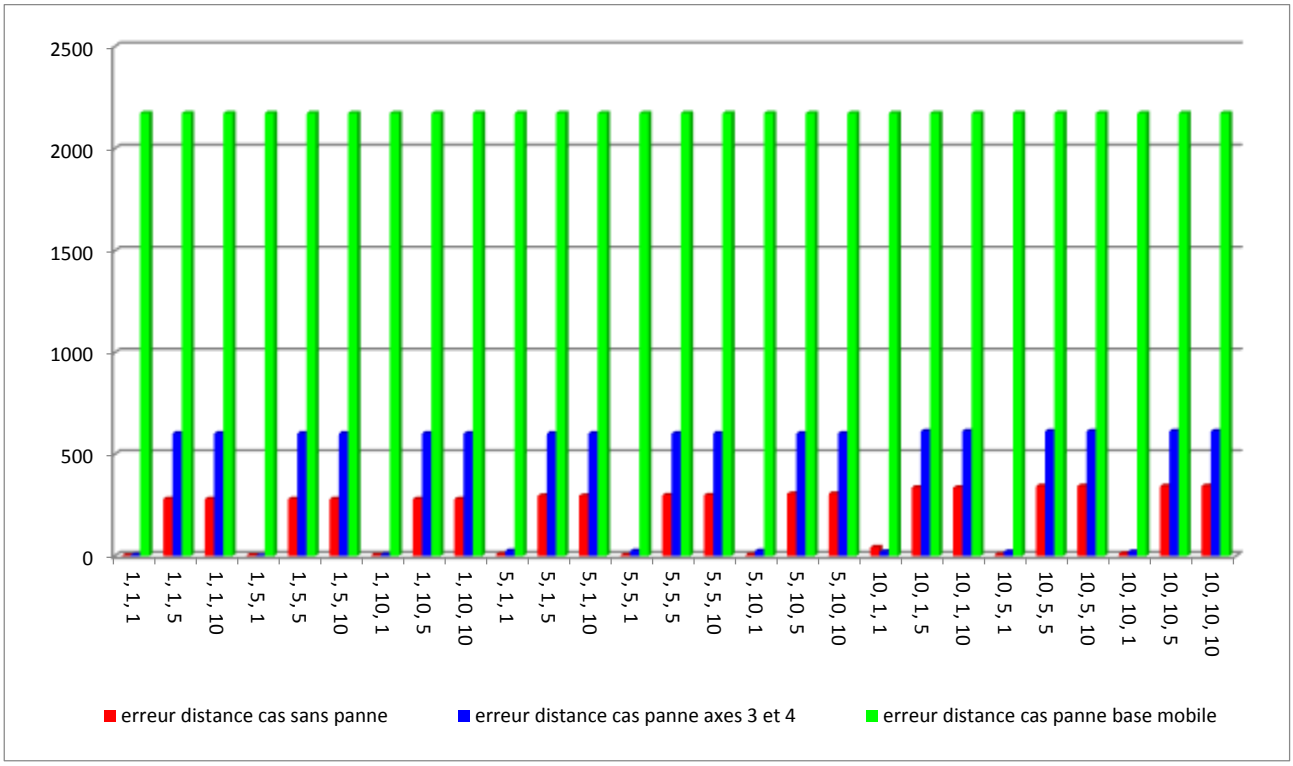


(a) Erreurs en distance pour les 27 combinaisons de pas avec les trois cas de figures

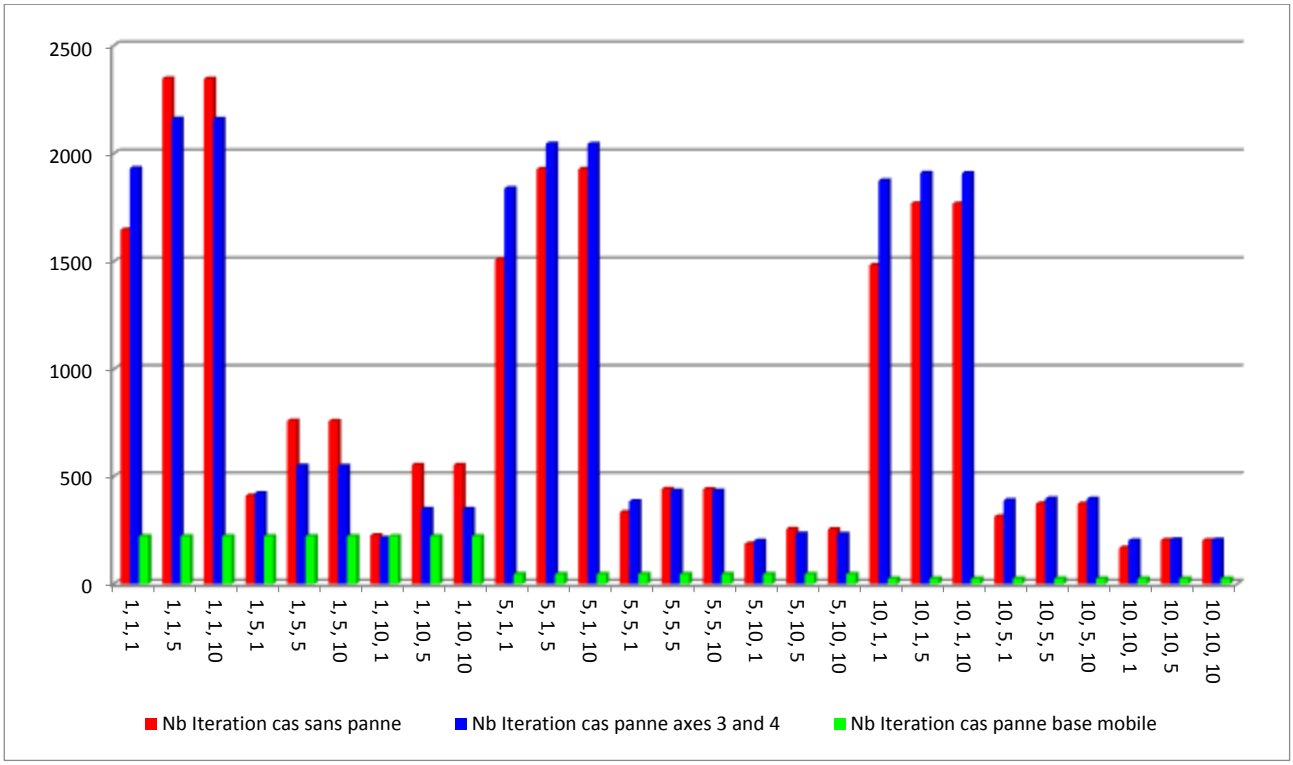


(b) Nombre d'itérations pour les 27 combinaisons de pas avec les trois cas de figures

Figure 46. Résultats obtenus pour le troisième scénario

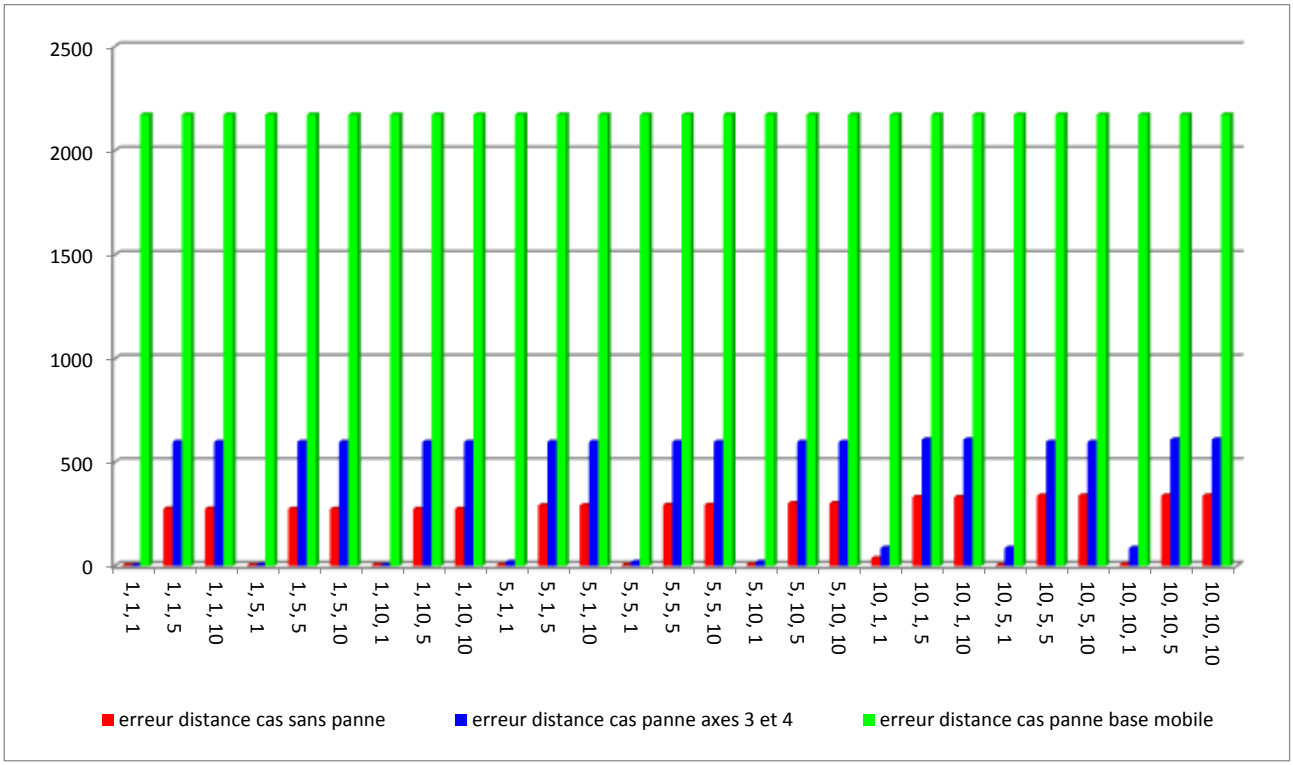


(a) Erreurs en distance pour les 27 combinaisons de pas avec les trois cas de figures

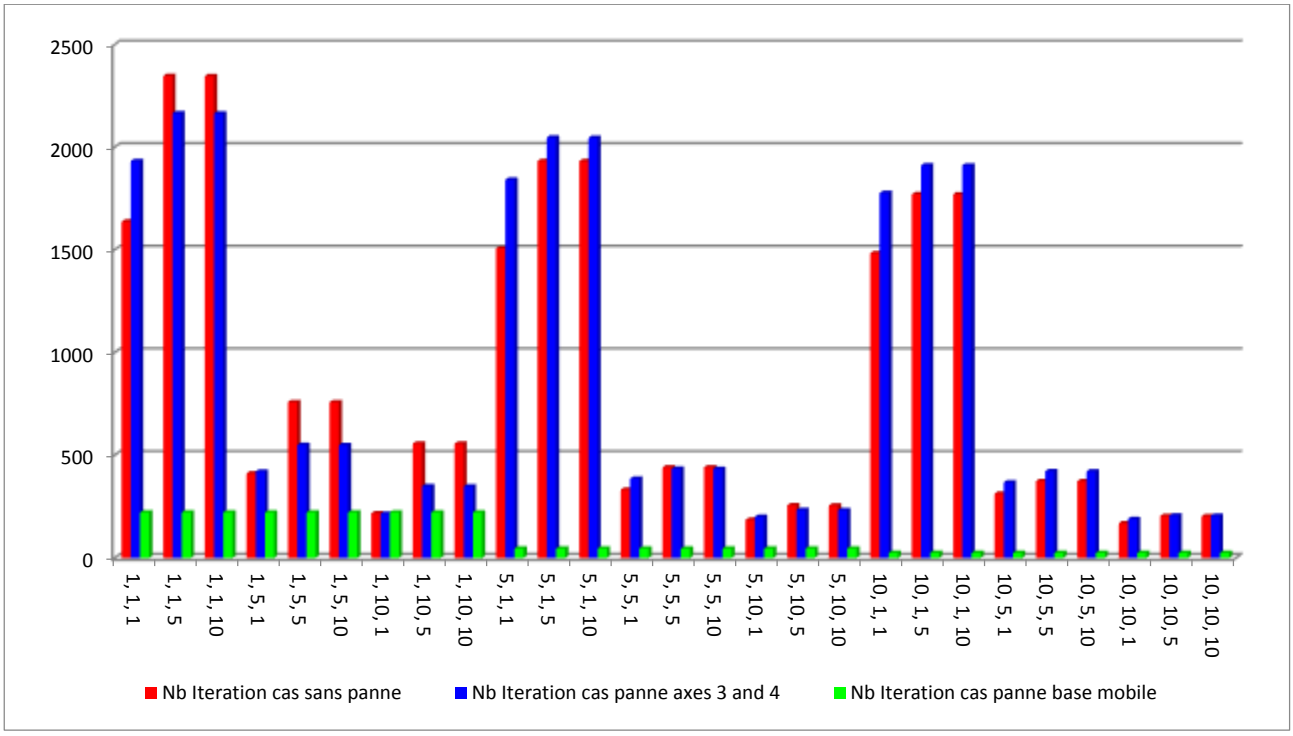


(b) Nombre d'itérations pour les 27 combinaisons de pas avec les trois cas de figures

Figure 47. Résultats obtenus pour le quatrième scénario



(a) Erreurs en distance pour les 27 combinaisons de pas avec les trois cas de figures

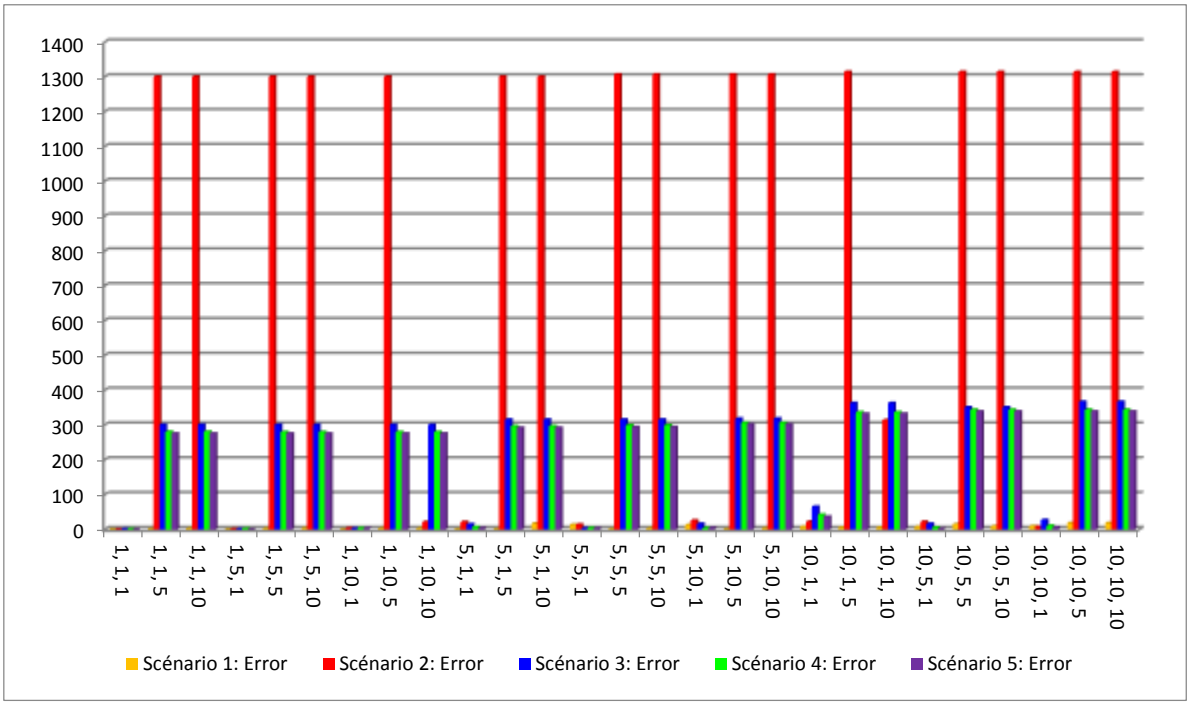


(b) Nombre d'itérations pour les 27 combinaisons de pas avec les trois cas de figures

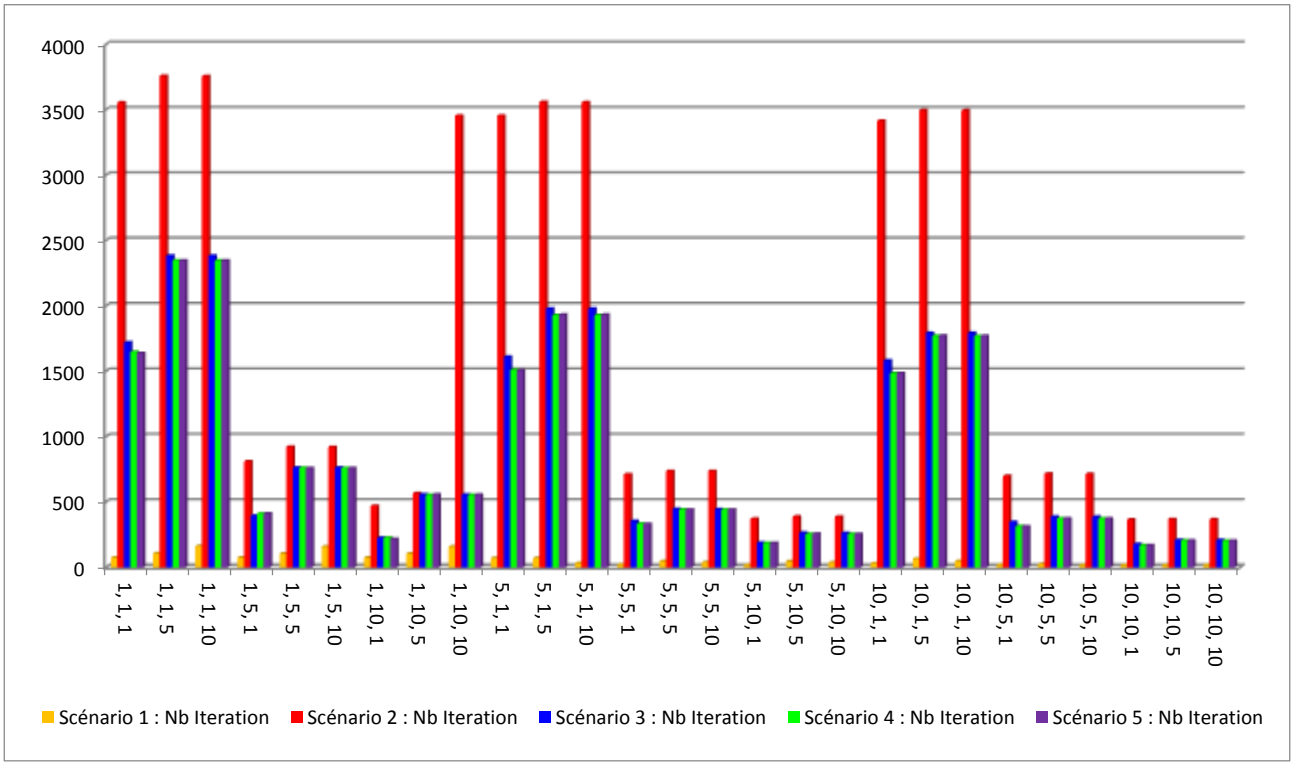
Figure 48. Résultats obtenus pour le dernier scénario

4.3.2. Synthèse des résultats obtenus

Dans cette section, nous allons synthétiser les résultats obtenus à travers des histogrammes présentant les deux critères considérées (*i*) *erreur en distance* et (*ii*) *nombre d'itérations*. Ceci a été élaboré dans le but d'étudier l'attitude d'une même combinaison de pas face à des scénarios différents pour les trois cas de simulation examinés (*i*) sans panne, (*ii*) avec une panne au niveau du manipulateur, et finalement (*iii*) la panne de la base mobile.

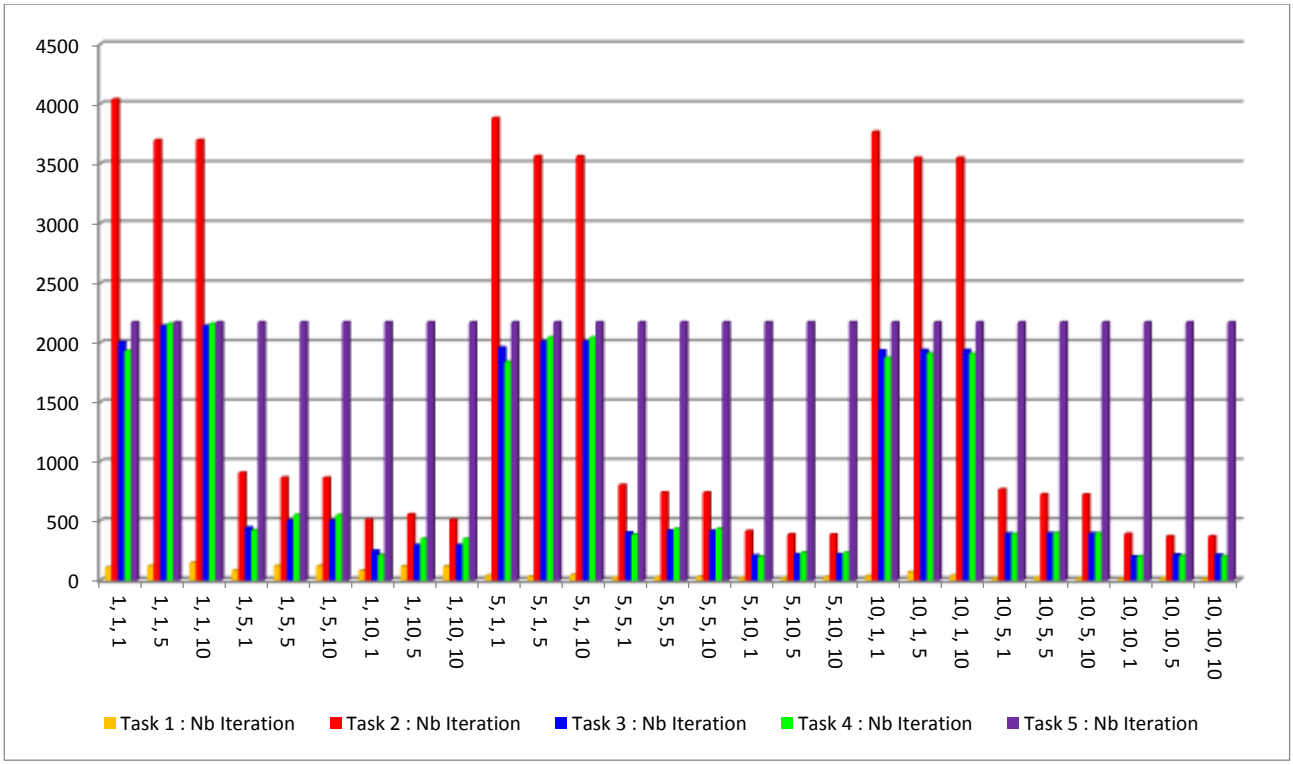


(a) Erreur en distance pour les cinq scénarios et avec les 27 combinaisons de pas

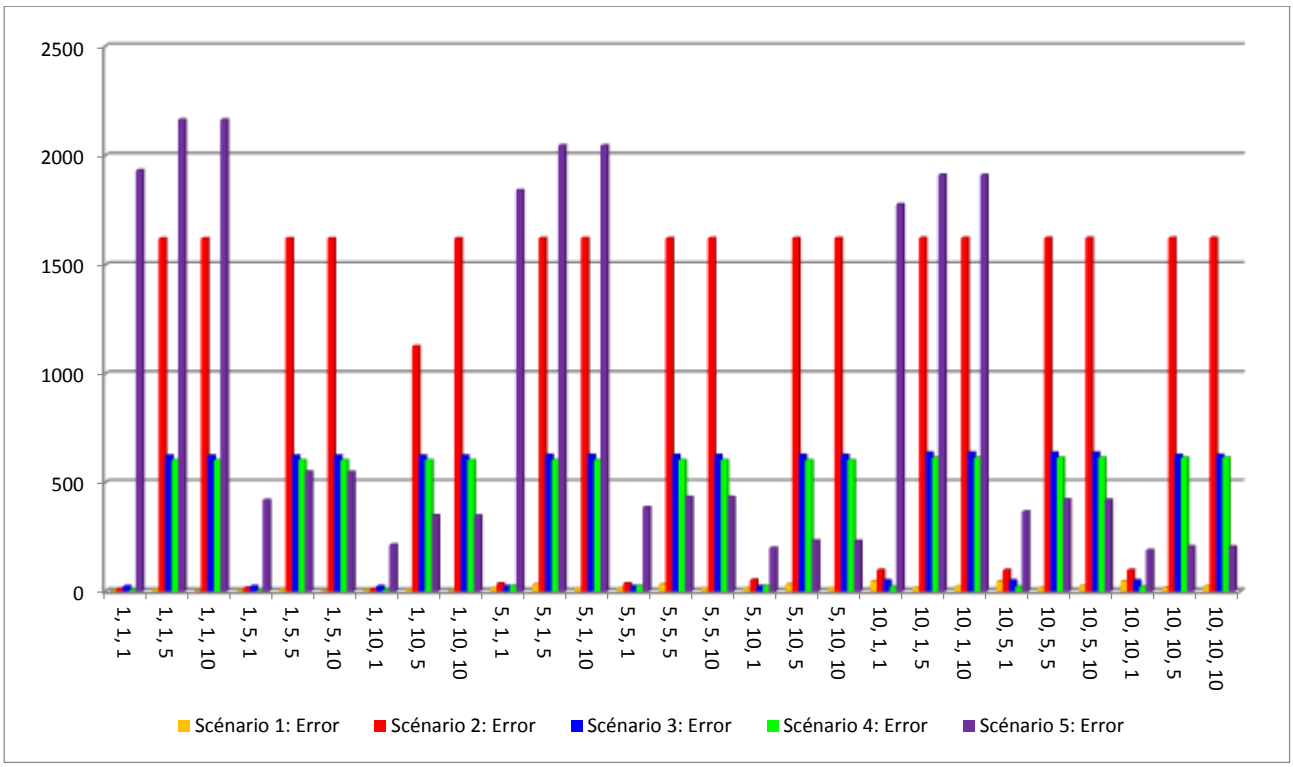


(b) Nombre d'itérations pour les cinq scénarios et avec les 27 combinaisons de pas

Figure 49. Résultats obtenus dans le cas sans panne

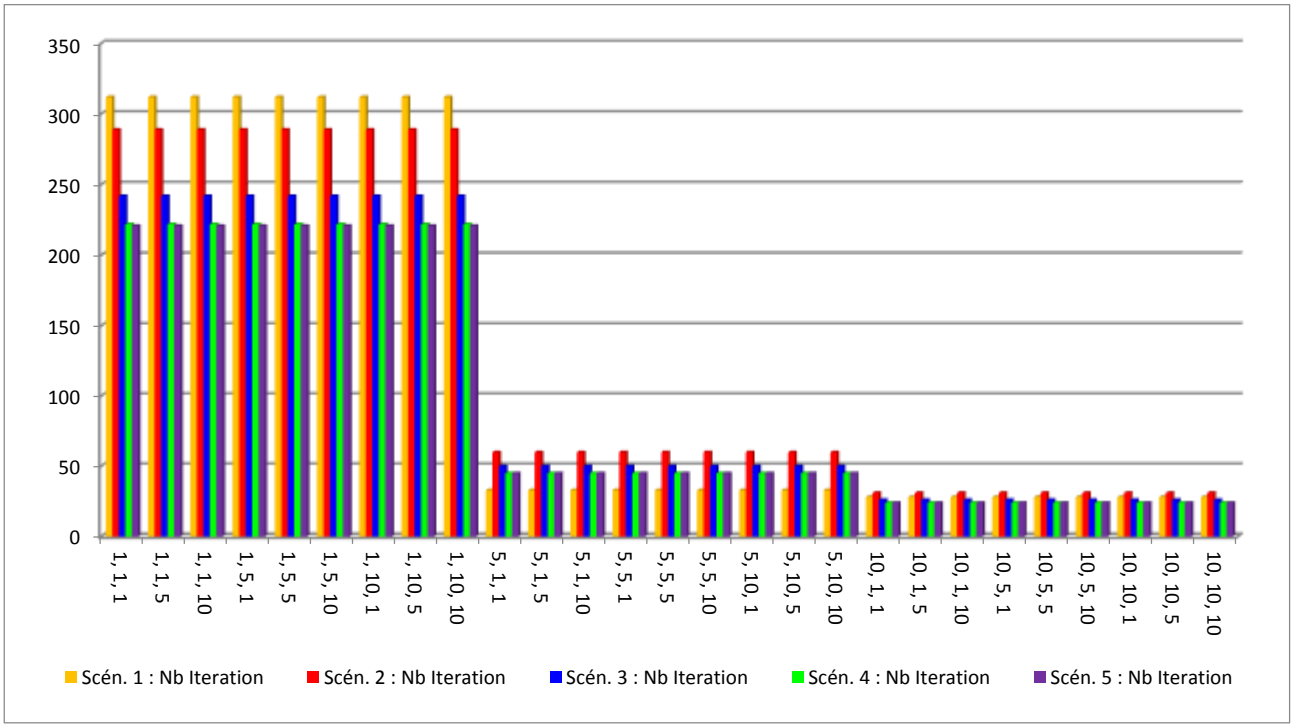


(a) Erreur en distance pour les cinq scénarios et avec les 27 combinaisons de pas

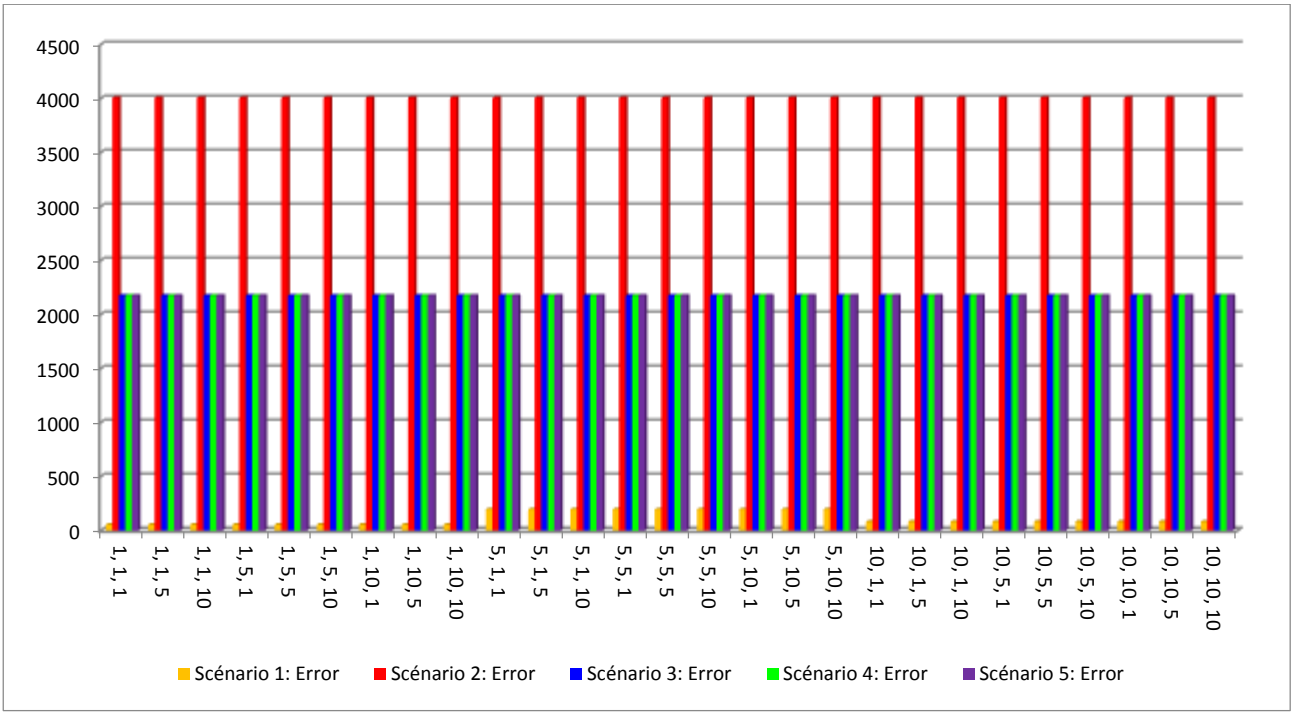


(b) Nombre d'itérations pour les cinq scénarios et avec les 27 combinaisons de pas

Figure 50. Résultats obtenus dans le cas de la panne de quelques articulations du manipulateur



(a) Erreur en distance pour les cinq scénarios et avec les 27 combinaisons de pas



(b) Nombre d'itérations pour les cinq scénarios et avec les 27 combinaisons de pas

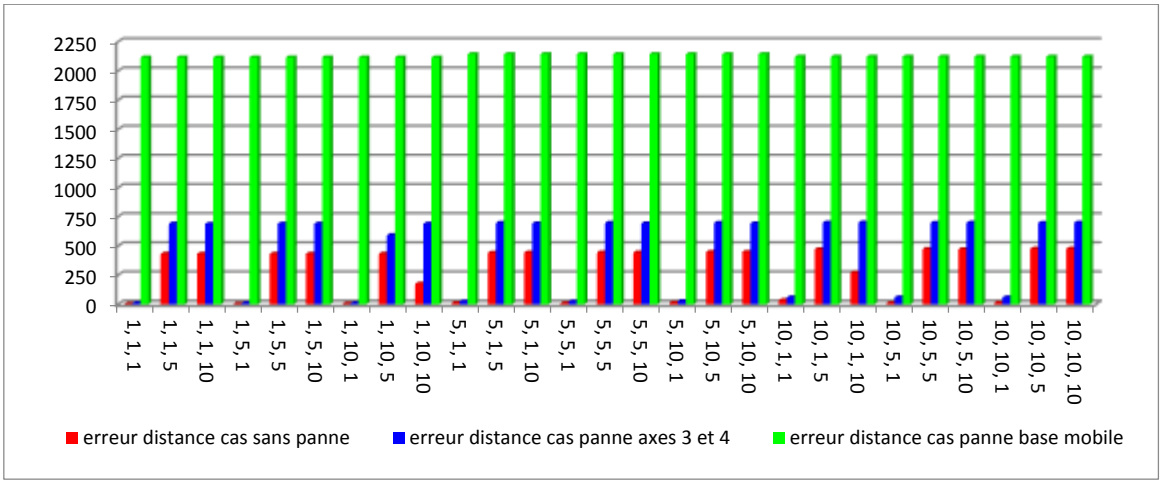
Figure 51. Résultats obtenus dans le cas de la panne de la base mobile

En étudiant de près les diagrammes présentés précédemment, nous pouvons constater que les différentes combinaisons que nous avons utilisées ont donné des résultats différents. Certaines compositions minimisent l'erreur en distance et permettent d'avoir des positions très proches de la cible imposée. D'autres propositions arrivent à la fin d'exécution en un nombre d'itérations relativement réduit.

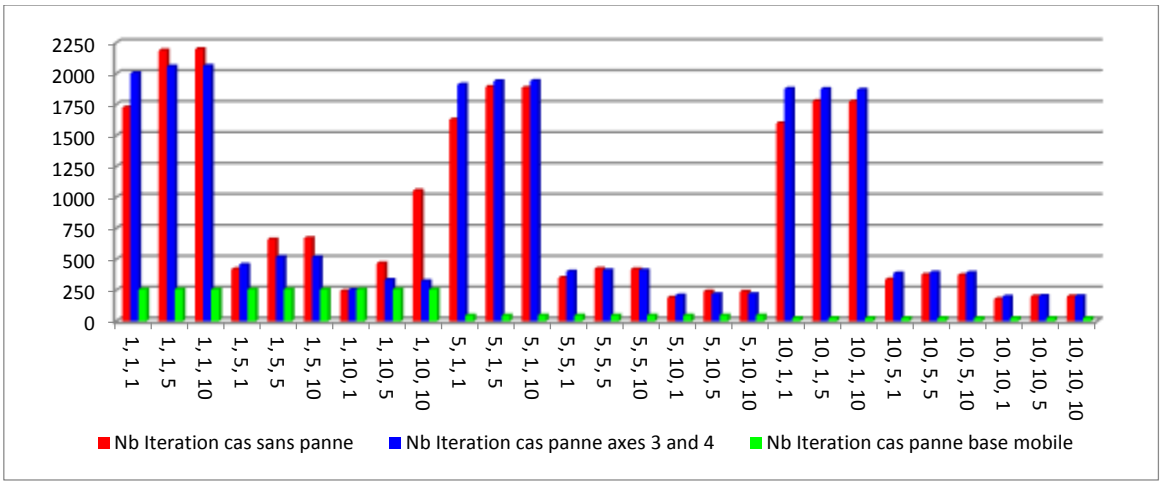
Il faut noter que les résultats d'un seul diagramme ne permettent pas d'évaluer l'efficacité d'une combinaison de pas. Puisque une combinaison *i* qui a donné de meilleurs résultats pour un scénario donné, ne donne pas forcément des bons résultats pour un autre scénario. Pour cette raison, une étude plus approfondie s'avère nécessaire pour la détermination de la meilleure combinaison.

4.3.3. Détermination de la meilleure combinaison de pas (les *Footsteps*)

Puisque une et une seule combinaison de pas n'est permise lors de l'exécution de l'approche, il faut déterminer la bonne. De ce fait et afin de choisir les pas les optimaux, nous avons classé les cinq meilleurs résultats trouvés pour chaque scénario, selon les deux critères d'évaluation précédents (*i*) *erreur minimale*, et (*ii*) *nombre d'itérations*. Les informations récoltées nous ont permis de dresser plusieurs tableaux de comparaison. Nous présentons le contenu de ces tableaux dans la section qui suit, afin de choisir la meilleure combinaison des pas.



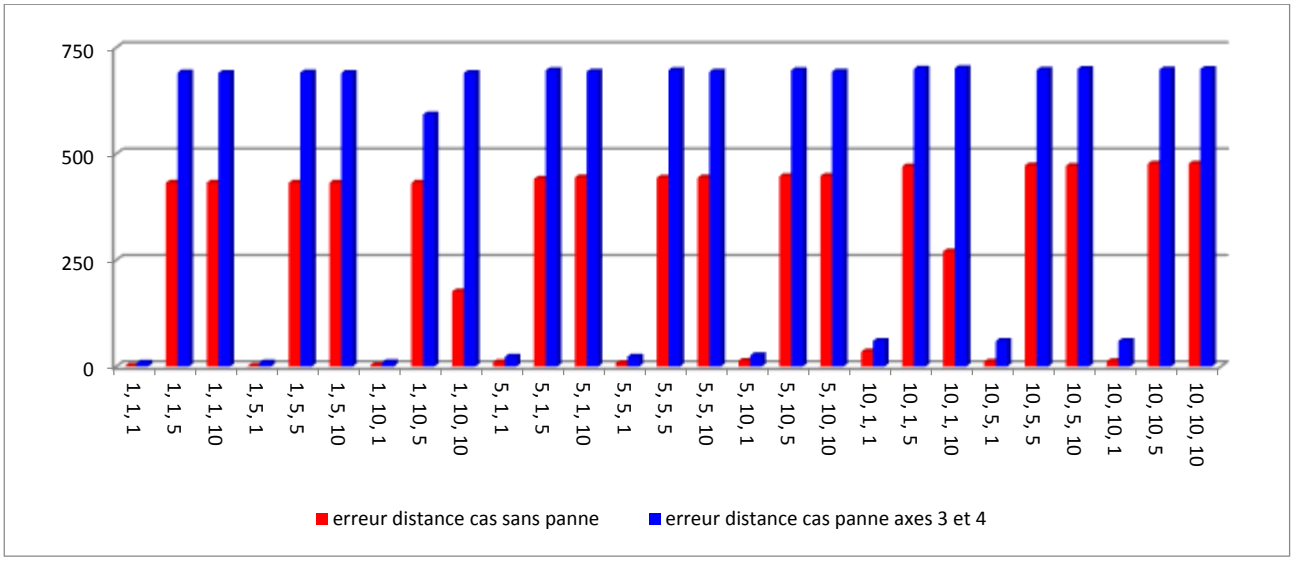
(a) Erreur moyenne par combinaison de pas pour les trois cas considérés



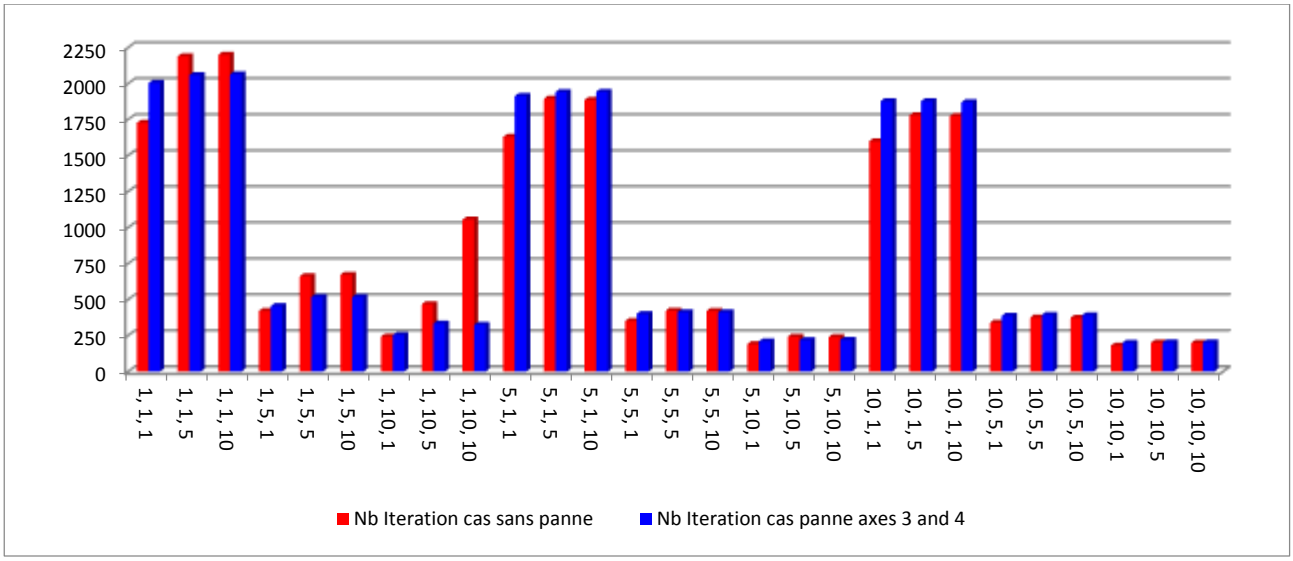
(b) Nombres moyens d'itérations par combinaison de pas pour les trois cas considérés

Figure 52. Erreur moyenne et nombre moyen d'itérations par combinaison de pas(1)

Pour la recherche du meilleur pas, nous n'allons considérer que le premier et le deuxième cas de figure, c.-à-d. le cas sans panne et le cas avec la panne des axes 3 et 4 du manipulateur. Pour le dernier cas (panne de la base mobile), toutes les solutions ont été acceptées (solutions sous-optimales). Ce dernier cas n'est pas significatif et ne peut être considéré lors de la détermination du meilleur pas pour la recherche de la solution optimale. Les deux figures qui suivent représentent, respectivement, les erreurs moyennes en distance et le nombre d'itérations.



(a) Erreur moyenne par combinaison de pas pour le premier et le deuxième cas



(b) Nombres moyens d'itérations par combinaison de pas pour premier et le deuxième cas

Figure 53. Erreur moyenne et nombre moyen d'itérations par combinaison de pas(2)

4.3.3.1. Détermination du meilleur pas pour le mouvement des articulations du manipulateur (*Joint footstep*)

Le tableau suivant donne le meilleur pas pour le mouvement des axes du manipulateur (*Joint footstep*) :

Meilleure solution : Sans panne + panne axes 3 et 4								
Pas groupés	**Pas**	**Tâche 1**	**Tâche 2**	**Tâche 3**	**Tâche 4**	**Tâche 5**	**Sous Total**	**Total**
1, x, x	1, 1, x	6	2	2	2	1	13	34
	1, 5, x	2	2	2	2	3	11	
	1, 10, x	2	2	2	2	2	10	
5, x, x	5, 1, x	2	1	2	0	1	6	17
	5, 5, x	1	2	2	1	2	8	
	5, 10, x	1	0	1	1	0	3	

10, x, x	10, 1, x	0	0	1	1	0	2	04
	10, 5, x	0	0	0	1	0	1	
	10, 10, x	0	1	0	0	0	1	

Tableau 2. Détermination du meilleur pas (Joint footstep) pour le mouvement des axes

Il est clair que le meilleur pas est celui qui génère le maximum de solutions optimales (meilleures) pour tous les problèmes traités. Dans notre cas, *Joint footstep*=1° (*meilleur pas=1, x, x*). Par la suite, il y a lieu de déterminer le meilleur *Base Translation footstep* et le meilleur *Base Rotation footstep*.

4.3.3.2. Détermination du meilleur pas pour le mouvement en translation de la base mobile (*Base Translation footstep*)

Le tableau suivant donne les meilleurs pas pour le mouvement en translation de la base mobile (*Base Translation footstep*) :

Meilleure solution : Sans panne + panne axes 3 et 4								
Pas groupés	**Pas**	**Tâche 1**	**Tâche 2**	**Tâche 3**	**Tâche 4**	**Tâche 5**	**Sous Total**	**Total**
x, 1, x	1, 1, x	6	2	2	2	2	14	21
	5, 1, x	2	1	2	0	1	5	
	10, 1, x	0	0	1	1	0	2	
x, 5, x	1, 5, x	2	2	2	2	2	10	20
	5, 5, x	1	2	2	1	2	8	
	10, 5, x	0	0	0	1	1	2	
x, 10, x	1, 10, x	2	2	2	2	2	10	14
	5, 10, x	1	0	1	1	0	3	
	10, 10, x	0	1	0	0	0	1	

Tableau 3. Détermination du meilleur pas (Base Translation footstep) pour le mouvement en translation de la base mobile

Comme pour le cas précédent, le meilleur pas est celui qui génère le maximum de solutions meilleures pour tous les problèmes considérés. Dans ce cas, nous avons sélectionné deux pas différents, puisque ils ont généré un nombre de résultats qui se rapproche. 21 solutions optimales pour le premier et 20 solutions pour le deuxième. Le premier est *Base Translation footstep=1mm* (x, 1, x) et le deuxième *Base Translation footstep=5mm* (x, 5, x).

4.3.3.3. Détermination du meilleur pas pour le mouvement en rotation de la base mobile (*Base Rotation footstep*)

Dans cette sous-section, il y a lieu de déterminer le meilleur pas pour *Base Rotation footstep*. Le tableau suivant résume les meilleurs pas pour le mouvement en rotation de la base mobile (*Base Rotation footstep*) :

Meilleure solution : Sans panne + panne axes 3 et 4								
Pas groupés	**Pas**	**Tâche 1**	**Tâche 2**	**Tâche 3**	**Tâche 4**	**Tâche 5**	**Sous Total**	**Total**
x, x, 1	x, 1, 1	3	3	5	3	3	17	47
	x, 5, 1	1	4	4	4	5	18	
	x, 10, 1	1	3	3	3	2	12	
x, x, 5	x, 1, 5	3	0	0	0	0	3	5
	x, 5, 5	1	0	0	0	0	1	
	x, 10, 5	1	0	0	0	0	1	
x, x, 10	x, 1, 10	2	0	0	0	0	2	4
	x, 5, 10	1	0	0	0	0	1	
	x, 10, 10	1	0	0	0	0	1	

Tableau 4. Détermination du meilleur pas (*Base Rotation footstep*) pour le mouvement en rotation de la base mobile

Comme pour les deux cas précédents, le meilleur pas est celui qui génère le maximum de solutions meilleures pour tous les problèmes considérés. Dans ce cas, le choix du pas est trivial. *Base Rotation footstep=1°* (47 meilleures solutions générées).

4.3.3.4. Récapitulation de la recherche de la meilleure combinaison de pas

Nous avons retenu trois combinaisons de pas pour la recherche de la solution optimale (1, x, x), (x, 1, x) et (x, 5, x), (x, x, 1). Cela permet de générer deux combinaisons différentes de pas pour la recherche de la meilleure solution pour tous les problèmes types considérés dans ce travail. Ils sont donnés comme suit :

- (*Joint footstep*, *Base Translation footstep*, *Base Rotation footstep*) = (1, 1, 1).
- (*Joint footstep*, *Base Translation footstep*, *Base Rotation footstep*) = (1, 5, 1).

Pour le choix final de l'un de ces deux pas, il est nécessaire d'adopter un autre critère de comparaison. Nous allons comparer leurs nombres d'itérations respectives lors de la recherche des meilleures solutions :

Nombre d'itérations : Sans panne + panne axes 3 et 4						
Pas		**Scénario 1**	**Scénario 2**	**Scénario 3**	**Scénario 4**	**Scénario 5**
1, 1, 1	Sans panne	75	3555	1719	1647	1638
	Avec panne	110	4039	2004	1932	1933
1, 5, 1	Sans panne	75	810	395	410	411
	Avec panne	85	905	444	422	420

Tableau 5. Sélection du meilleur pas entre (1, 1, 1) et (1, 5, 1)

Il y a lieu de constater que la recherche de la meilleure solution en utilisant la première combinaison de pas, c.-à-d. (1, 1, 1), nécessite un nombre d'itérations très élevé et, par conséquent, un temps d'exécution plus important en la comparant avec l'autre combinaison de pas (1, 5, 1). Cette dernière est retenue comme étant la combinaison optimale de pas pour la recherche de la meilleure solution.

Dans la suite de ce chapitre, les paramètres de simulation pour les différents scénarios sont donnés dans le tableau suivant :

Agent	Action	Footstep
Agents Axe	MoveUp	Joint footstep = 1°
	MoveDown	
Agent base mobile	MoveForward	Base Translation footstep = 5mm
	MoveBackward	
	TurnRight	Base Rotation footstep = 1°
	TurnLeft	

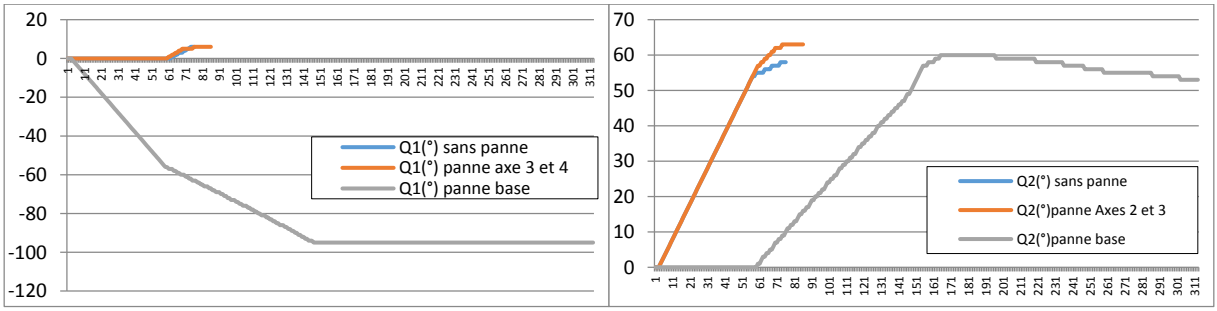
Tableau 6. Paramètres de simulation choisis pour les différents agents de contrôle

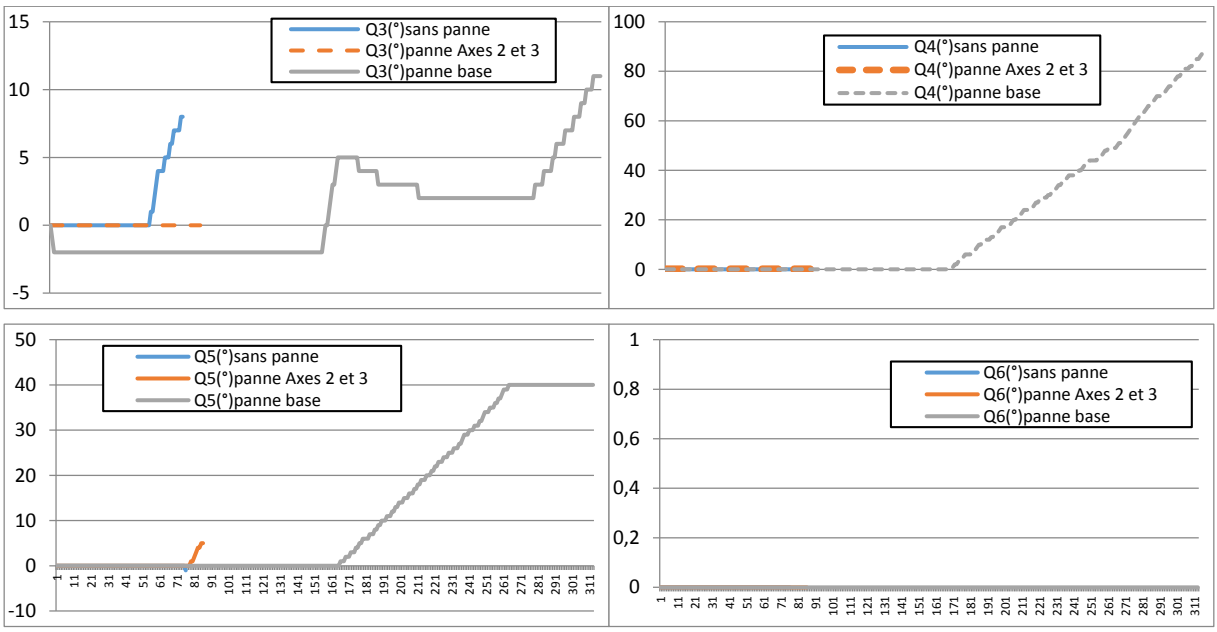
4.4. Résultats de simulation

Après avoir déterminé la meilleure combinaison de pas pour la recherche de la solution optimale, c.-à-d. (1, 5, 1), nous allons étudier, dans ce qui suit, les performances de l'approche proposée en examinant les fichiers trace résultants de la simulation, en considérant les trois cas traités. Nous étudierons, en premier lieu, les variations des variables articulaires du manipulateur. Par la suite, nous présenterons les variations du positionnement de la base mobile dans son environnement. Puis, nous examinerons les variations de l'erreur en distance, entre la position actuelle de l'effecteur et celle de la cible imposée. Enfin, les courbes indiquant les variations de la position courante de l'effecteur en 3D seront exposés. Les figures qui suivent montrent les courbes résultantes de cette analyse respectivement pour les cinq scénarios considérés dans ce qui précèdent.

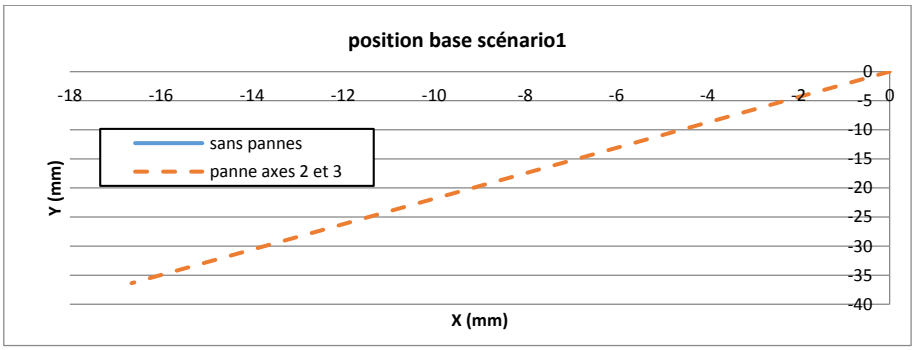
4.4.1. Résultats du premier scénario

Les figures qui se suivent montrent les résultats obtenus pour le premier scénario considéré.

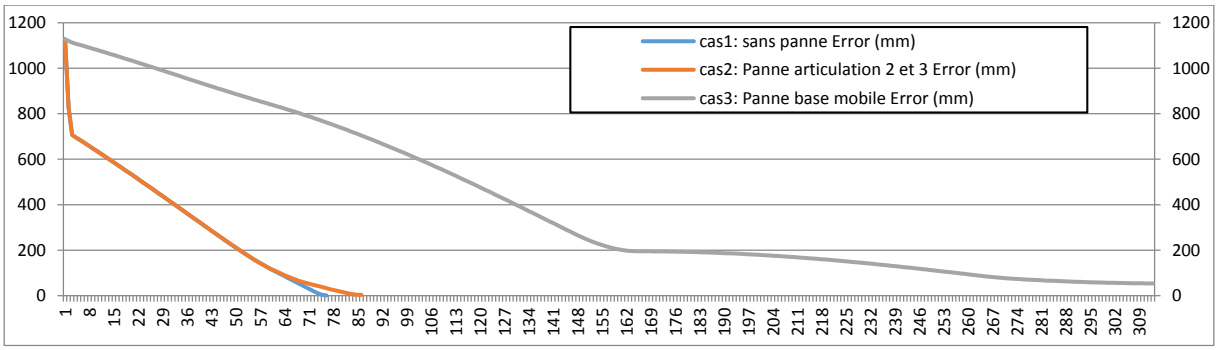




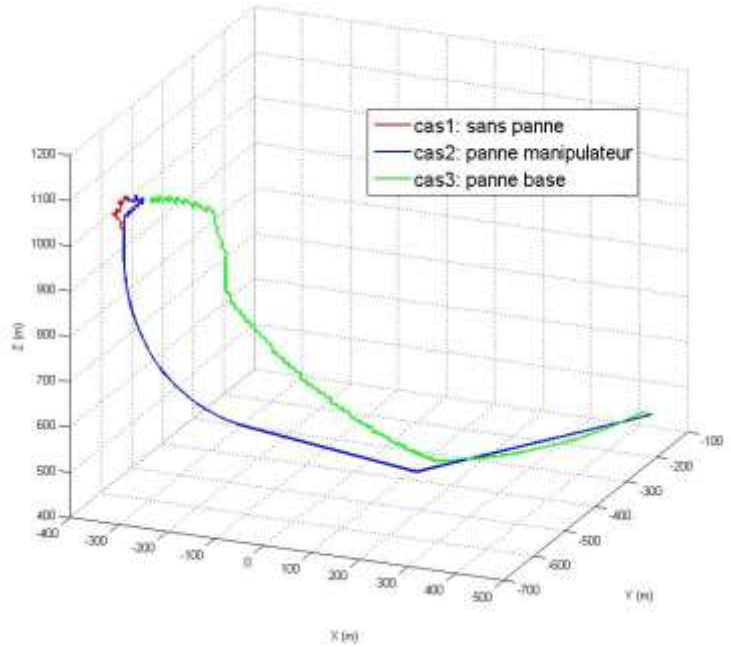
(a) Variations des positions articulaires du manipulateur



(b) Variations des coordonnées cartésiennes de la base mobile



(c) Variations de l'erreur en position de l'effecteur

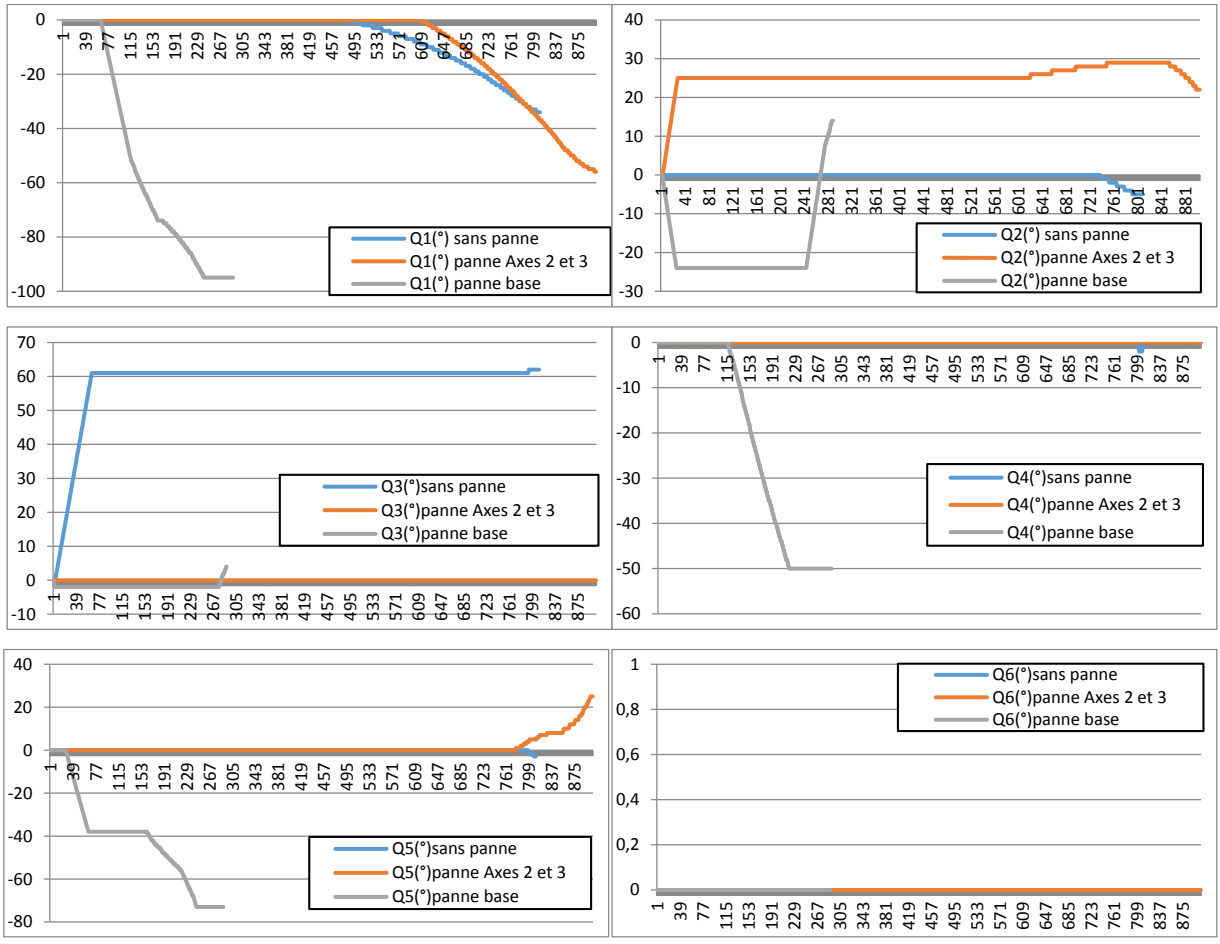


(d) Mouvement 3D de l'effecteur

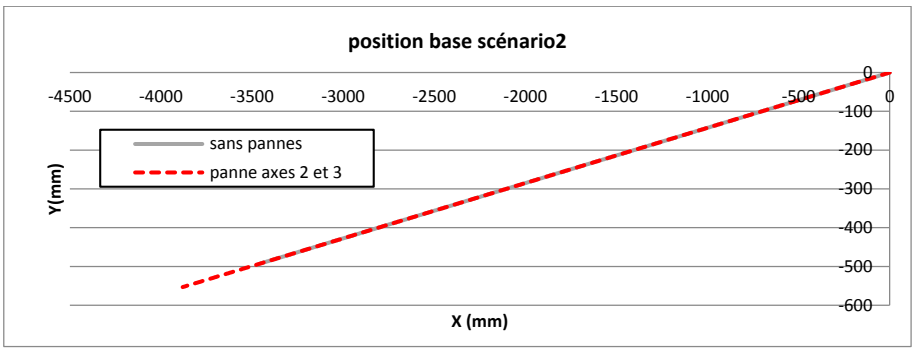
Figure 54. Résultats obtenus pour le premier scénario

4.4.2. Résultats du deuxième scénario

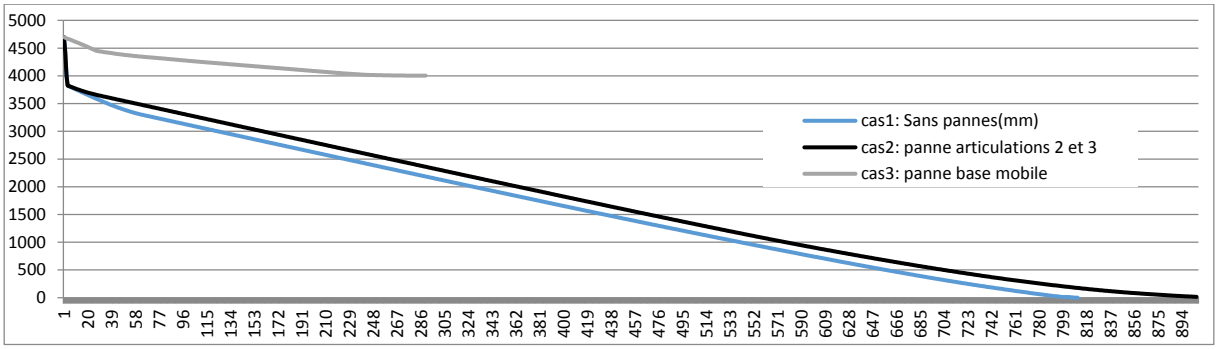
Les digrammes dans ce qui suit représentent les détails liés aux résultats dans le 2ème scénario traité.



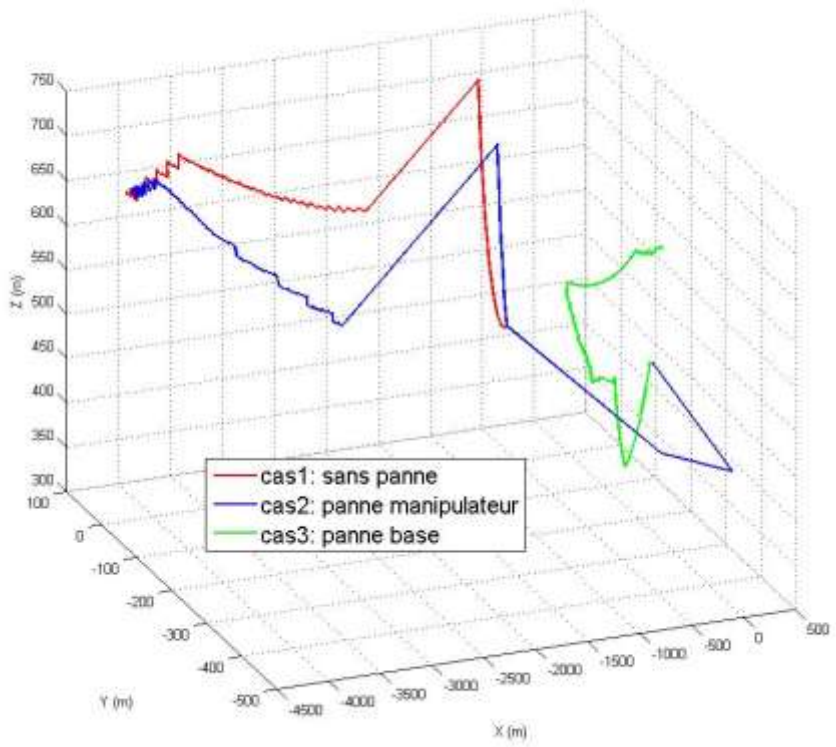
(a) Variations des positions articulaires du manipulateur



(b) Variations des coordonnées cartésiennes de la base mobile



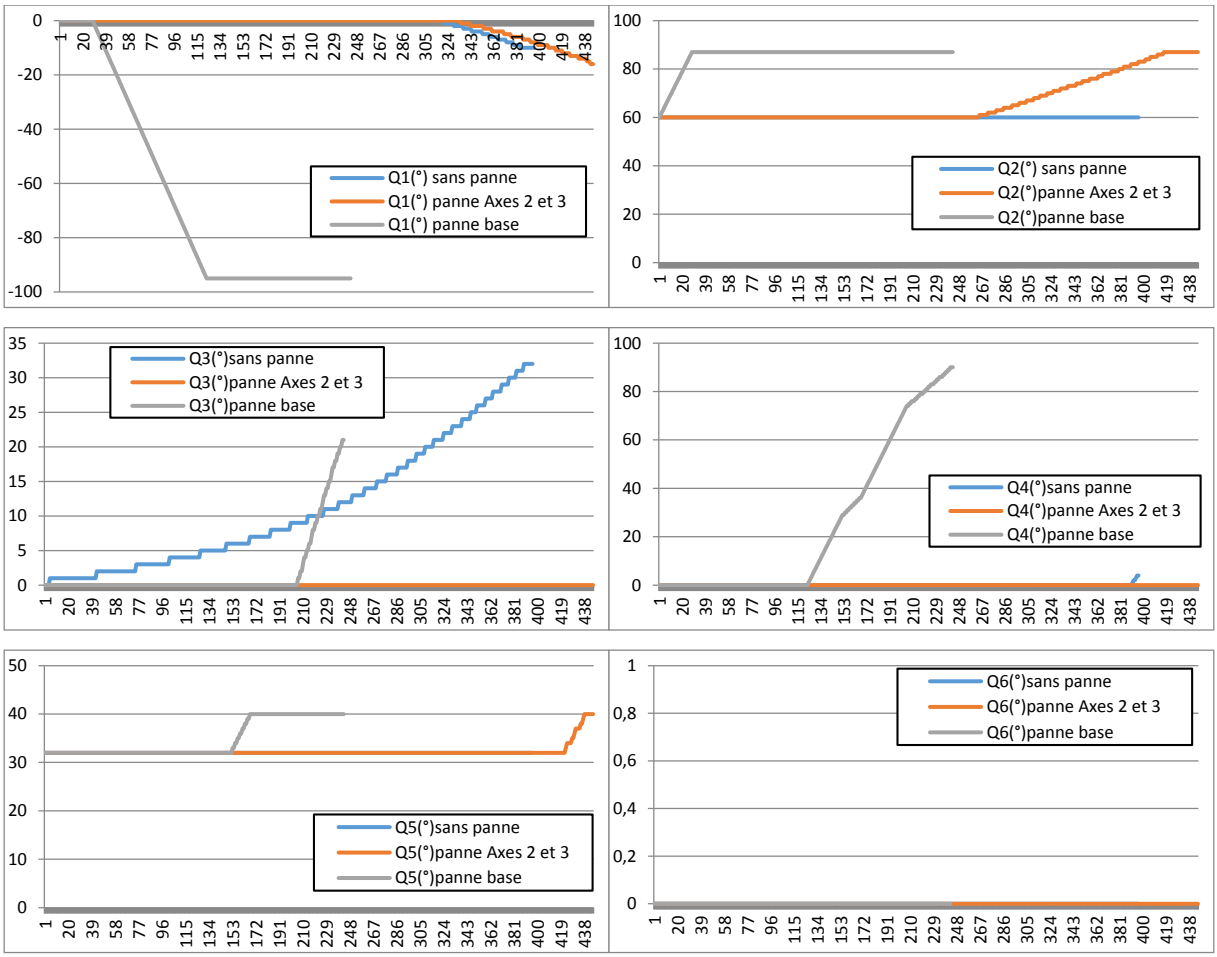
(c) Variations de l'erreur de positionnement de l'effecteur



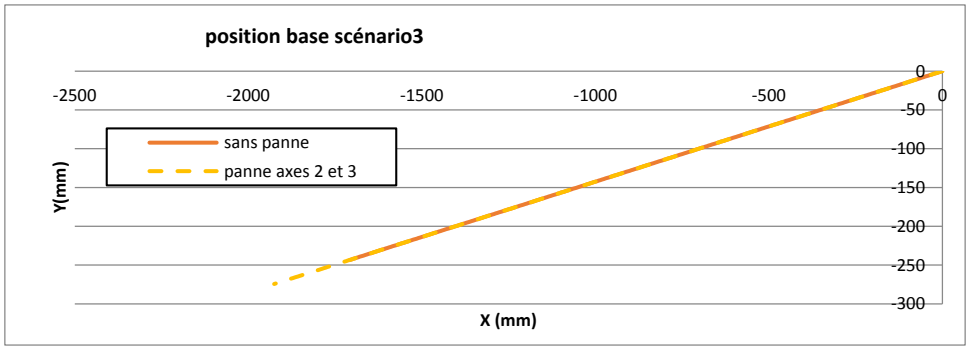
(d) Mouvement 3D de l'effecteur

Figure 55. Résultats obtenus pour le deuxième scénario

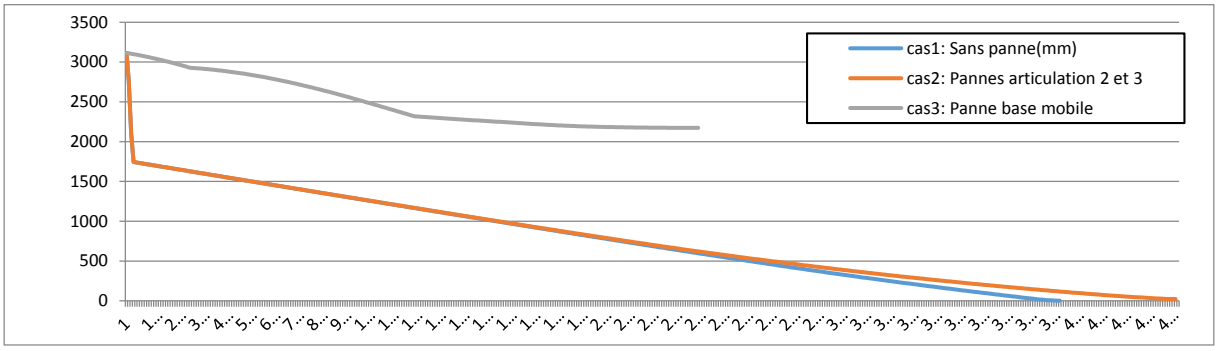
4.4.3. Résultats du troisième scénario



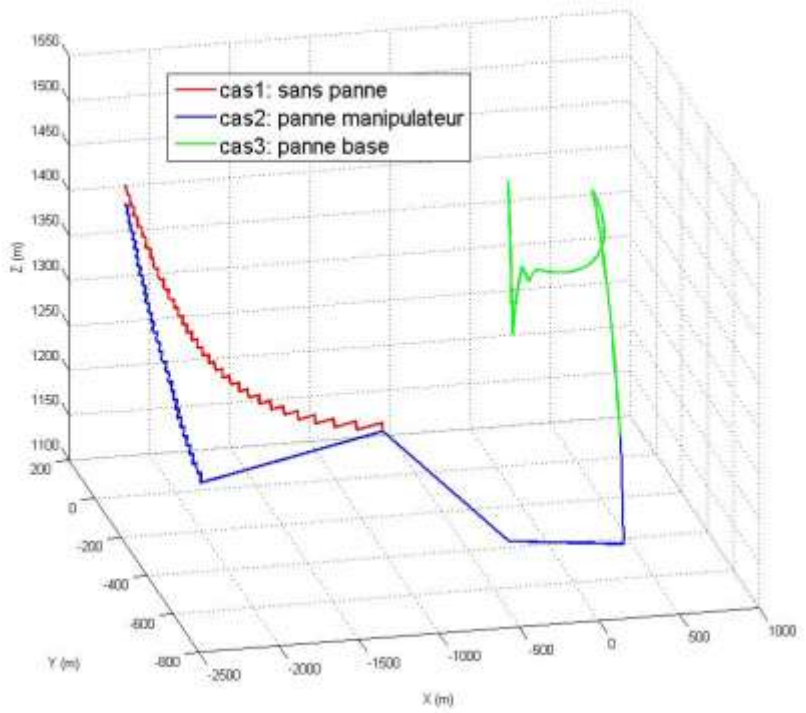
(a) Variations des positions articulaires du manipulateur



(b) Variations des coordonnées cartésiennes de la base mobile



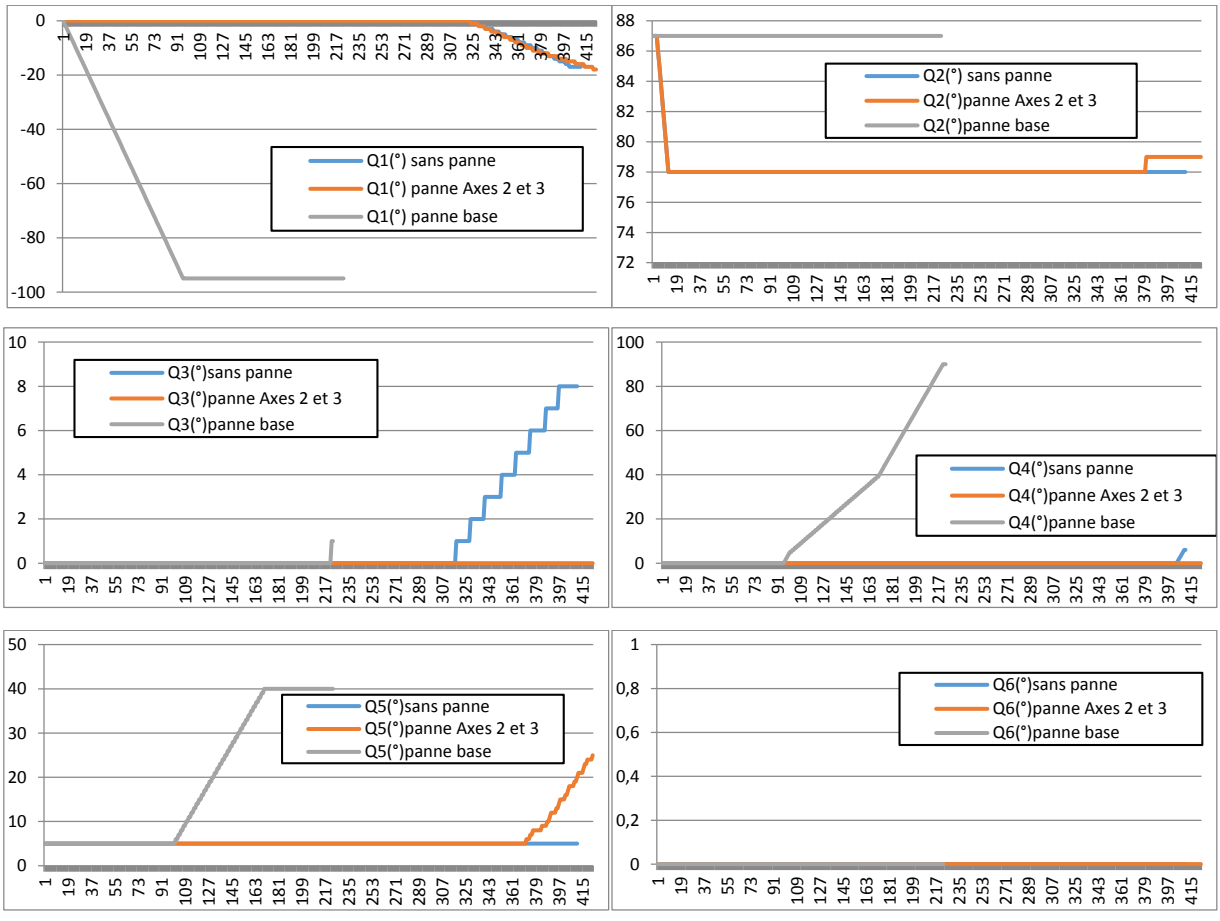
(c) Variations de l'erreur en positionnement de l'effecteur



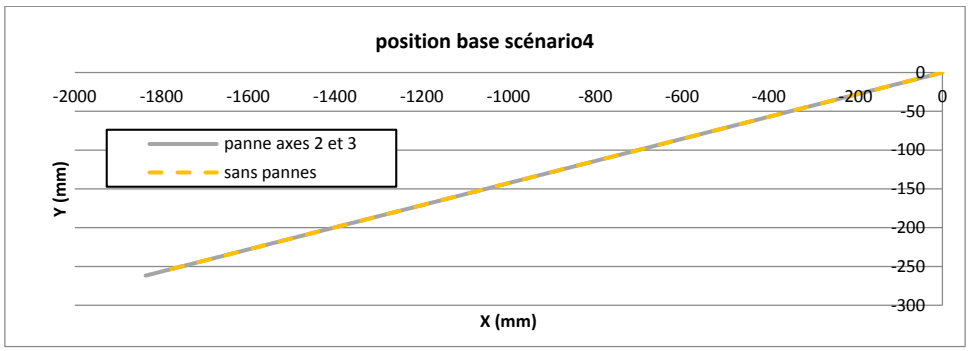
(d) Mouvement 3D de l'effecteur

Figure 56. Résultats obtenus pour le troisième scénario

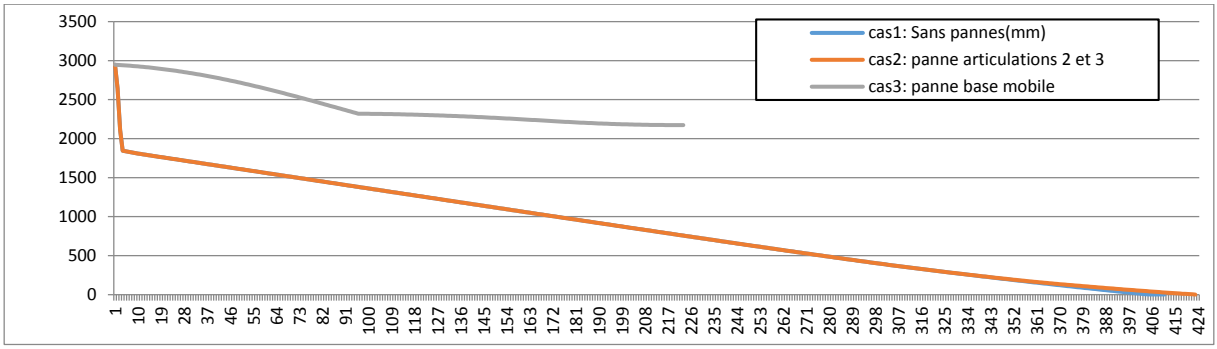
4.4.4. Résultats du quatrième scénario



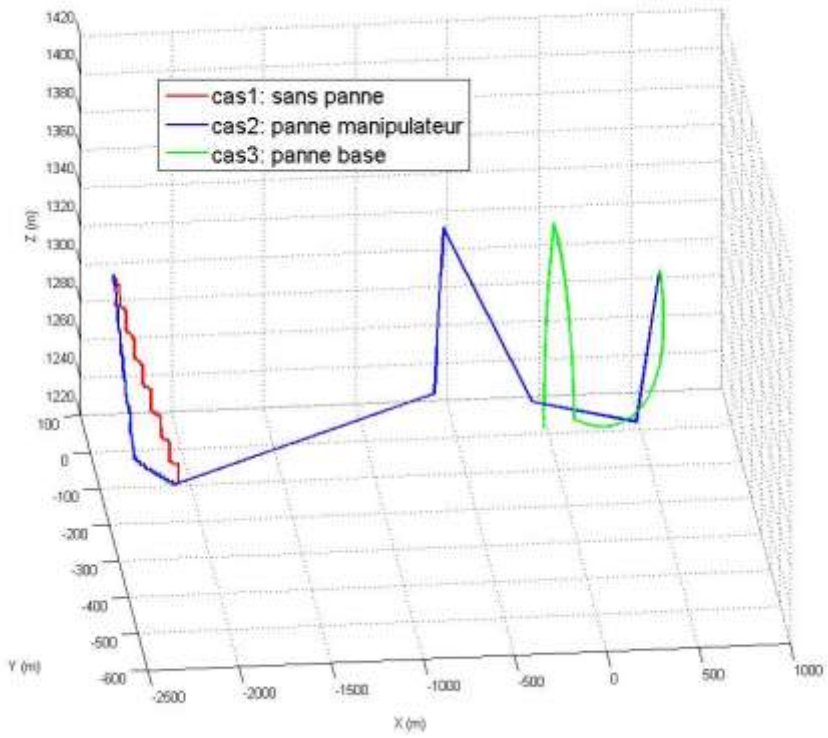
(a) Variations des positions articulaires du manipulateur



(b) Variations des coordonnées cartésiennes de la base mobile



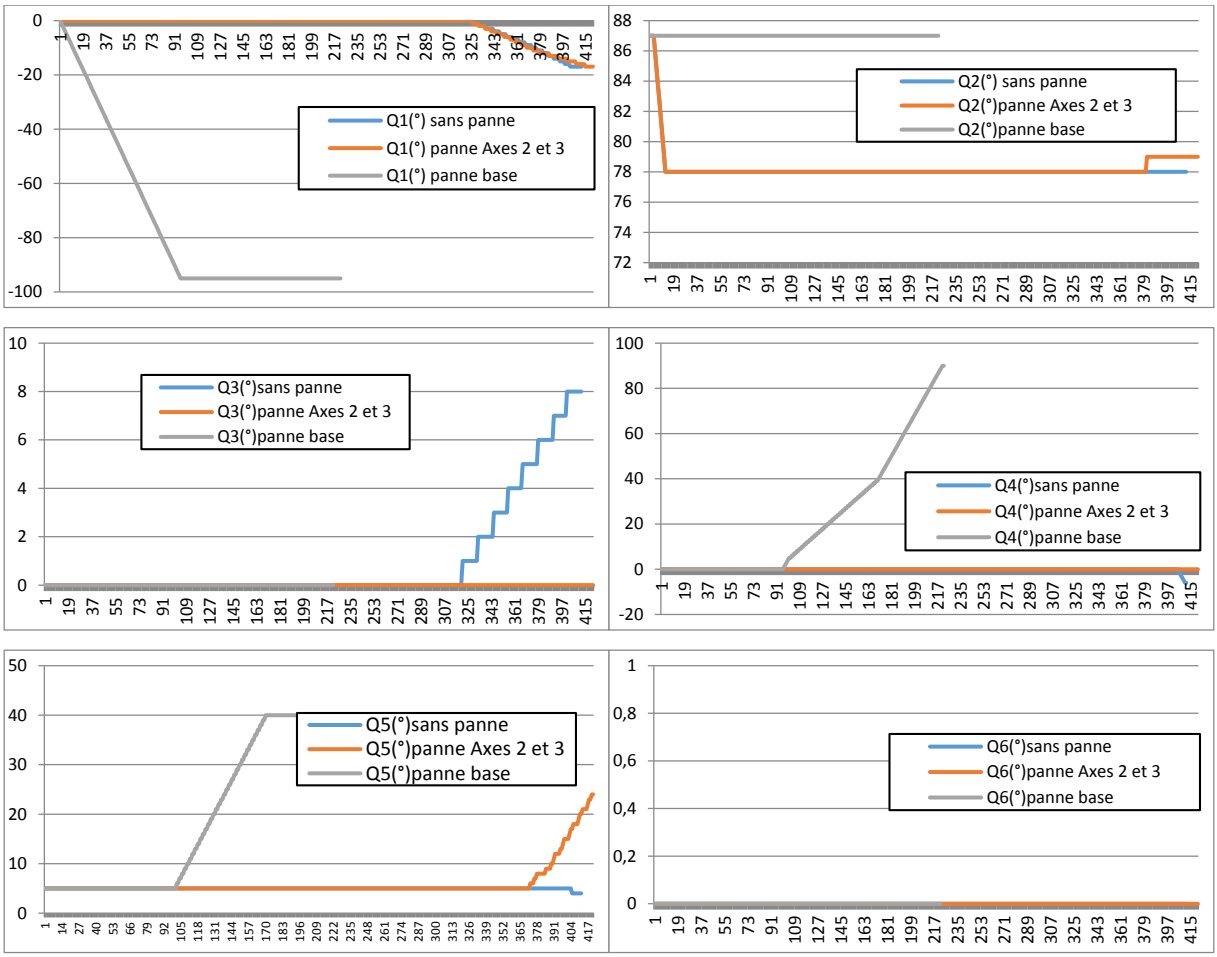
(c) Variations de l'erreur en position de l'effecteur



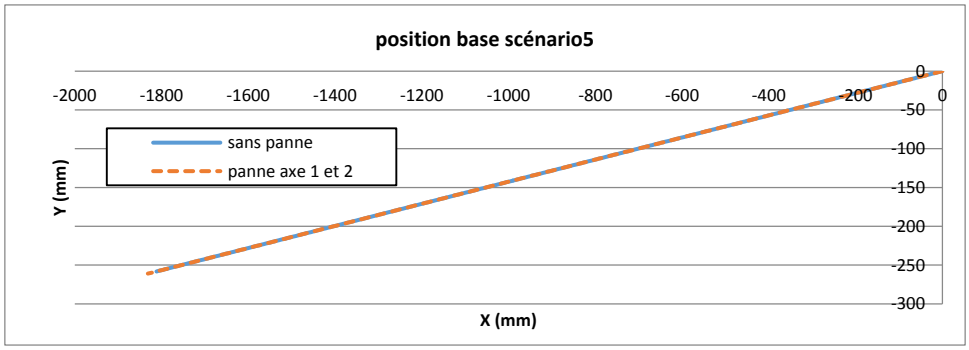
(d) Mouvement 3D de l'effecteur

Figure 57. Résultats obtenus pour le quatrième scénario

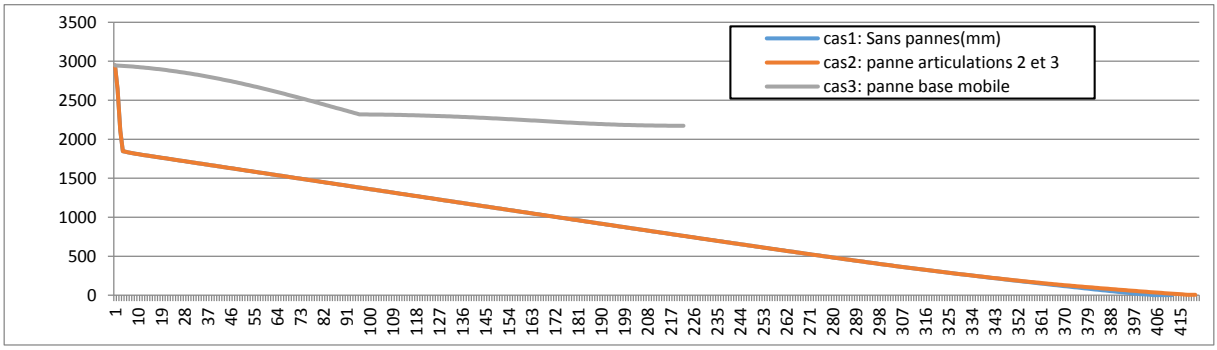
4.4.5. Résultats du cinquième scénario



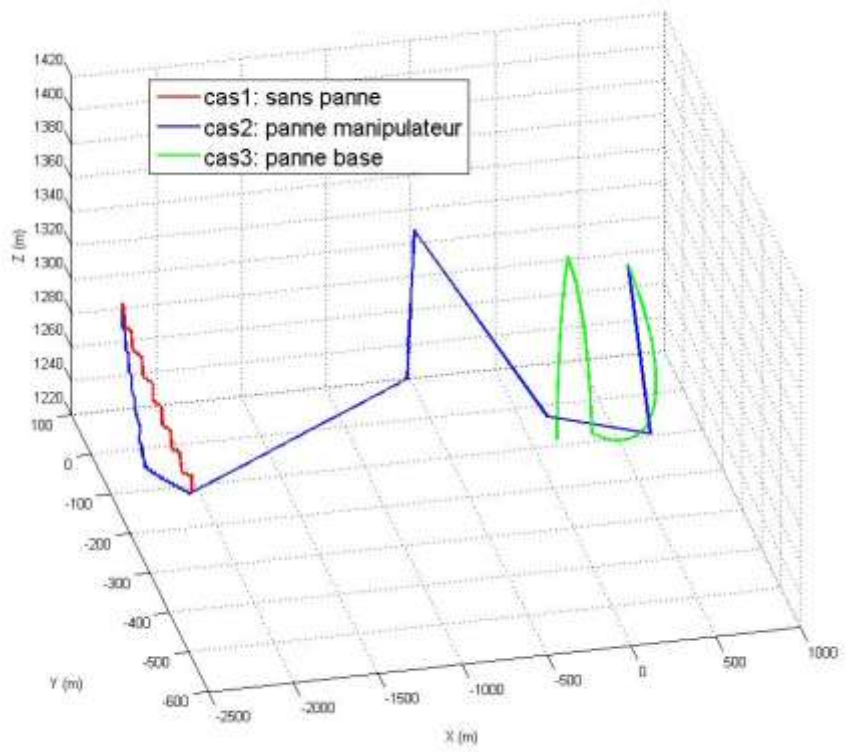
(a) Variations des positions articulaires du manipulateur



(b) Variations des coordonnées cartésiennes de la base mobile



(c) Variations de l'erreur de positionnement de l'effecteur



(d) Mouvement 3D de l'effecteur

Figure 58. Résultats obtenus pour le dernier scénario

4.4.6. Discussion des résultats

En analysant les courbes présentées ci-dessus, nous pouvons constater que l'approche que nous proposant donne de très bons résultats. À travers les différents scénarios considérés, le système de contrôle a pu satisfaire les besoins demandés, en minimisant la valeur de l'erreur en distance qui sépare la position de l'effecteur et celle de la cible imposée. Il a même pu offrir des résultats très satisfaisants dans le deuxième cas de simulation invoqué, où la présence d'une panne au niveau des articulations 3et 4 était signalée. Dans ce cas, le contrôleur a pu mener l'effecteur vers une position ci-proche de la position désirée, malgré la panne des deux articulations.

Le troisième cas de simulation invoqué, où nous avons simulé la tomber en panne de la base mobile, montre que l'approche proposée offre un service minimum même en cas d'anomalies extrêmes. Le système dans ce cas de figure essaye de rapprocher le maximum la position de l'effecteur avec celle de la cible, sans avoir besoin d'un mécanisme de recouvrement dédié.

4.5. Étude comparative

Afin d'offrir un support de comparaison pour les résultats de notre approche, nous allons présenter les résultats d'une approche de contrôle qui se basent sur le calcul des modèles

géométriques inverses (MGI) pour calculer les consignes de contrôle du robot. Les scénarios considérés dans ce qui précèdent ont été testés et évalués dans [28]. Dans ce travail, le processus de contrôle est effectué en deux étapes séquentielles (*i*) Déplacement de la base mobile vers une nouvelle situation proche de la position de la cible imposée, (*ii*) Calcul de la configuration du manipulateur permettant d'atteindre la position imposée, en se basant sur l'étude du MGI.

Il y a lieu de préciser que cette approche classique ne prend pas en considération la présence d'anomalies ou un changement dans la structure mécanique du robot. Donc les résultats présentés ci-dessous vont être comparés avec les résultats pour le cas sans de pannes seulement.

4.5.1. Cas sans pannes

Le tableau suivant récapitule les résultats obtenus en utilisant l'approche de contrôle proposée et celles basée sur le calcul du MGI pour tous les scénarios considérés dans ce travail. Le critère d'évaluation est l'erreur en distance qui sépare la position finale de l'effecteur et celle de la cible imposée.

		Approche classique basée sur le calcul du MGI	**Nouvelle approche proposée basée sur un SMA**
Scénario 01	$(x_B, y_B, \theta_B)_{Fin}$	(0, 0, 0)	(0, 0, -2)
	$(Q_1, \ldots, Q_6)_{Fin}$	(60, 61, 30, 95, -15, 0)	(6, 58, 8, 0, -1, 0)
	Erreur de distance	4.8689	0.7374
Scénario 02	$(x_B, y_B, \theta_B)_{Fin}$	(-3440, 13, 12)	(-3455.07, -492.50, -3)
	$(Q_1, \ldots, Q_6)_{Fin}$	(20, 32, 28, 0, …, 0)	(-34, -5, 62, 0, -3, 0)
	Erreur de distance	5.4928	1.3767
Scénario 03	$(x_B, y_B, \theta_B)_{Fin}$	(-1920, 2, 15)	(-1707.73, -243.43, -3)
	$(Q_1, \ldots, Q_6)_{Fin}$	(37, 52, 61, 73, -52, 28)	(-10, 60, 32, 4, 32, 0)
	Erreur de distance	6.67	1.4549
Scénario 04	$(x_B, y_B, \theta_B)_{Fin}$	(-1670, 0, 0)	(-1811.68, -258.24, -3)
	$(Q_1, \ldots, Q_6)_{Fin}$	(5, 49, 63, -13, -22, -78)	(-17, 78, 8, 6, 5, 0)
	Erreur de distance	12.3714	2.3908
Scénario 05	$(x_B, y_B, \theta_B)_{Fin}$	(-1560, 0, 0)	(-1811.68, -258.24, -3)
	$(Q_1, \ldots, Q_6)_{Fin}$	(5, 44, 69, -12, -23, -79)	(-17, 78, 8, -6, 4, 0)
	Erreur de distance	33.762	1.1338

Tableau 7. Comparaison entre une approche classique basée MGI et notre approche basée SMA

On constate que l'approche que nous avons proposée offre plus de précision en ce qui concerne la tâche considérée. Les erreurs en distances obtenues étaient meilleures dans les cinq scénarios considérés.

4.5.2. Cas de la panne des articulations 3 et 4

Dans ce deuxième cas, nous simulons la présence d'une panne au niveau des articulations 3 et 4 (à la position 0°). Cette panne survient au début du traitement et elle persiste pendant l'exécution des tâches considérées. Le tableau ci-après représente les résultats d'exécution pour les cinq scénarios ainsi que la situation finale de la base mobile et la configuration finale du manipulateur. Les critères d'évaluations sont l'erreur en distance ainsi que le nombre d'itérations. Il faut noter que l'approche classique, montrée précédemment, ne prend pas en considération les pannes au niveau du manipulateur.

Scénario 1	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	(-18.45, -11.96, 10°)
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(41, 59, **0, 0**, 17, 0)
	Erreur de distance (mm)	0.9664
	Nombre d'itérations	148
Scénario 2	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	(-3880.77, -553.19, -3°)
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-56, 22, **0, 0**, 25, 0)
	Erreur de distance (mm)	15.7449
	Nombre d'itérations	905
Scénario 3	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	(-1925.53, -274.47, -3)
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-16, 87, **0, 0**, 40, 0)
	Erreur de distance (mm)	22.6522
	Nombre d'itérations	444
Scénario 4	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	(-1836.43, -261.77, -3°)
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-18, 79, **0, 0**, 25, 0)
	Erreur de distance (mm)	0.3878
	Nombre d'itérations	422
Scénario 5	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	(-1810.69, -258.10, -3°)
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-17, 73, **0, 0**, 40, 0)
	Erreur de distance (mm)	2.1747
	Nombre d'itérations	1933

Tableau 8. Résultats obtenus par l'approche proposée dans le cas d'une panne des axes 3 et 4 du manipulateur

4.5.3. Cas de la panne de la base mobile

De même que pour le cas précédent, nous simulons ici la présence d'une panne au niveau de la base mobile. Cette panne survient à l'instant t=0 (au début de l'exécution) et persiste durant toute l'exécution. La base mobile reste dans sa situation initiale (x_{base}, y_{base}, $theta_{base}$) = (0, 0, 0). Le

tableau suivant montre les résultats de simulation obtenus. Comme pour le deuxième cas considéré, l'approche roboticienne ne prend pas en considération la panne de la base mobile.

Scénario 1	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	**(0, 0, 0)**
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-95, 53, 11, 88, 40, 0)
	Erreur de distance (mm)	54.0627
	Nombre d'itérations	312
Scénario 2	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	**(0, 0, 0)**
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-95, 14, 4, -50, -73, 0)
	Erreur de distance (mm)	4004.0195
	Nombre d'itérations	289
Scénario 3	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	**(0, 0, 0)**
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-95, 87, 21, 90,40, 0)
	Erreur de distance (mm)	2172.9592
	Nombre d'itérations	242
Scénario 4	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	**(0, 0, 0)**
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-95, 87, 1, 90, 40, 0)
	Erreur de distance (mm)	2173.7636
	Nombre d'itérations	222
Scénario 5	$Base_{Fin}(x_B, y_B, \theta_B)_{Fin}$	**(0, 0, 0)**
	$Configuration_{Fin}(q_1, q_2, q_3, q_4, q_5, q_6)_{Fin}$	(-95, 87, 0, 90, 40, 0)
	Erreur de distance (mm)	2173.0541
	Nombre d'itérations	221

Tableau 9. Résultats obtenus par l'approche proposée dans le cas d'une panne de la base mobile

4.6. Conclusion

Afin de valider le système de contrôle proposé, plusieurs scénarios de validations étaient considérés pour étudier son comportement face à des situations différentes. Nous avons considéré aussi, dans nos expérimentations, la réaction du contrôleur face à des situations imprévues, où l'apparition de différents types de pannes était simulée. Les résultats de simulation étaient très prometteurs.

Pour situer notre travail par rapport à la littérature, une étude comparative était mise en œuvre. Nous avons comparé, ainsi, les résultats de l'approche proposée avec celles d'un travail récent basé sur une approche roboticienne. Notre approche donne des résultats meilleurs par rapport au travail considéré. Il y a lieu de signaler que les résultats en question étaient seulement pour le premier cas que nous avons considéré, vu que les approches roboticiennes classiques ne prennent pas en considération un changement dans la structure mécanique du robot ou la présence de pannes.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

L'objectif principal de ce travail consiste à élaborer une architecture multi-agents de contrôle des manipulateurs mobiles ayant un seul manipulateur et évoluant dans un environnement à trois dimensions. La tâche de validation consiste à atteindre une situation finale imposée à son effecteur afin de délivrer un objet, sans et avec considération des défaillances de quelques articulations du manipulateur et sans et avec considération de la panne de la base mobile.

Selon la vocation et la nature des traitements, deux types d'agents ont été identifiés dans l'approche de contrôle proposée :

- Les *agents système* sont chargés d'assurer le traitement des données issues des différents capteurs possibles. Ils regroupent plusieurs fonctionnalités qui ont un lien direct avec la collecte de données qui rentrent dans le processus de contrôle.
- Les *agents de contrôle* sont dédiés à l'opération de contrôle. À chaque agent est affectée la tâche de contrôler un sous-système mécanique (base mobile ou une des articulations du manipulateur). Un seul agent hybride, *Agent Base mobile*, a été identifié pour le contrôle de la base mobile. Aussi, un agent réactif, *Agent Axe* est affecté au contrôle de chacune des articulations du manipulateur. Les consignes de contrôle issues de ce type d'agents sont calculées en utilisant un mécanisme de mouvement virtuel-vérification. Le contrôle adopté est un contrôle en position, c'est-à-dire, les sorties envoyées par chaque agent sont des consignes en positions.

Chacun des agents de contrôle requiert la position actuelle de l'effecteur ainsi que celle de la cible. Les coordonnées cartésiennes de ces deux points sont utilisées lors de l'estimation des erreurs. Dans ce travail, nous n'avons considéré que l'erreur en distance.

Chaque agent de contrôle essayera de faire coïncider la position de l'effecteur du robot avec celle de la cible imposée. Après chaque mouvement virtuel, l'agent calcule la nouvelle valeur de la fonction objective, *f_Obj*, entre la nouvelle situation de l'effecteur et celle la cible imposée. Après avoir évalué les mouvements possibles, l'agent de contrôle sélectionne la nouvelle situation du sous-système qu'il contrôle correspondant au mouvement qui minimise *f_Obj*. À la fin de chaque itération, les agents de contrôle envoient leurs *meilleurs choix*, ainsi que les valeurs de *f_Obj* à à autre agent hybride, en occurrence l'*agent Superviseur*. Ce dernier est, alors, chargée de sélectionner le meilleur mouvement (qui minimise l'erreur) à entreprendre.

Deux mouvements ont été identifiés pour chaque agent de ce type avec un pas fixe égal à *Joint_footstep* (*MoveUp* et *MoveDown*). De même, quatre types de mouvements élémentaires différents sont possibles pour l'*Agent Base mobile* (*MoveForward, MoveBackward, TurnRight* et *TurnLeft*) avec un pas fixe qui est égal à *BaseTranslationFootstep* pour les mouvements en translation et *BaseTranslationFootstep* pour les mouvements en rotation.

Nous nous somme basés sur la plateforme open source *jade* (*Java Agent DEvelopment framework*) comme outil d'implémentation du système multi-agents de contrôle. La plateforme *jade* présente un environnement de création et de manipulation des agents ainsi qu'un ensemble d'outils graphiques pour le monitoring et le débogage d'applications. Chaque agent *jade* est censé réaliser un traitement propre à lui d'une manière totalement indépendante. Cela n'est garanti qu'avec une exécution autonome pour chaque agent. Par conséquent, un comportement global, permettant la réalisation des tâches complexes, émerge de l'interaction entre les différents agents. Cette interaction est réalisée via un modèle de communication basé sur un échange de message. Ces messages ont un format précis, spécifié par le langage *ACL* (*Agent Communication Language*). Les types de message les plus usuels sont INFORM, REQUEST, PROPOSE, etc.

Pour des besoins de validation de l'approche proposée, nous avons développé une interface graphique pour la simulation des scénarios. Cette interface permet de lancer la simulation d'un ou plusieurs scénarios à la fois. Un scénario de validation est défini par :

- La position de la cible donnée par (X_{E_Final}, Y_{E_Final}, Z_{E_Final}).
- La situation initiale de la base mobile donnée par (X_{B_Init}, Y_{B_Init}, θ_{B_Init}).
- La configuration initiale du manipulateur donnée par (Q_{1_Init}, …, Q_{6_Init}).

Avant d'avoir étudié les performances de l'approche proposée en simulation, nous avons déterminé la meilleure combinaison de pas (*JointsFootstep*, *BaseTranslationFootstep* et

BaseRotationFootstep) pour la recherche de la solution optimale. Pour cela, nous avons considéré plusieurs combinaisons. Après plusieurs tentatives, nous avons fixé un intervalle entre 1 et 10, et plus précisément, nous avons opté pour l'ensemble {1, 5, 10} ce qui donne 27 combinaisons possibles. Afin de sélectionner une seule combinaison de pas, nous avons exécuté, pour chaque scénario, 27 instances avec toutes les combinaisons possibles. Par la suite, nous avons évalué les résultats obtenus en considérant l'*erreur minimale* et le *nombre d'itérations*. À la fin de cette procédure, la meilleure combinaison de pas a été déterminée avec succès. Cette combinaison est (*JointsFootstep*, *BaseTranslationFootstep*, *BaseRotationFootstep*)= (1°, 5mm, 1°).

Dans le but d'offrir un support de comparaison de l'approche proposée avec les autres approches de la littérature, cinq scénarios différents ont été choisi pour étudier son comportement face à des situations différentes. Chacun de ces scénarios est considéré pour tester une situation particulière ou pour valider un comportement spécifique. Nous avons considéré aussi la réaction du contrôleur face à des situations imprévues, où l'apparition de différents types de pannes était simulée.

L'architecture de contrôle a été validée avec succès sur plusieurs tâches. Les résultats obtenus, se sont avérés très satisfaisants et très prometteurs, comparés aux travaux existants dans la littérature. De plus, cette approche est très intéressante du fait qu'elle ne requiert pas de calculs mathématiques lourds et complexes, ce qui la rend plus adaptée au contrôle temps réel. L'exécution en parallèle, assurée par le paradigme multi-agents, minimise considérablement le temps de traitement global ce qui optimise le temps de réponse du système. Cette approche offre aussi un contrôle tolérant aux pannes qui peuvent surgir, et plus précisément, les changements inattendus dans la structure mécanique du robot. Cette spécificité a fourni un service minimum en cas d'anomalies liées au dysfonctionnement d'une ou plusieurs articulations du manipulateur ou de la base mobile.

Les performances et la robustesse de l'approche de contrôle proposée doivent être montrées et discutées à travers des exemples avec d'autres types de problèmes. Cela ouvre plusieurs perspectives sur des travaux futurs. On cite entre autres :

- Dans l'approche de contrôle proposée, nous avons considéré que les pas (les *footsteps*) sont fixes. Il serait, également, intéressant d'améliorer cette approche en implémentant des mécanismes permettant une sélection dynamique des pas utilisés (*JointsFootstep*, *BaseTranslationFootstep* et *BaseRotationFootstep*) par les agents de contrôle au cours de l'exécution. Cela, peut être réalisé en ayant recours aux techniques du soft-computing. Il est envisageable, dans notre cas, de développer un contrôleur basé sur la logique floue pour le choix dynamique des combinaisons de pas.

- D’autres techniques de soft-computing peuvent être intéressantes dans l’implémentation des agents hybrides. Nous citons comme exemple l’utilisant de la logique floue, les réseaux de neurones ou bien l’association de ces deux derniers (contrôle neuro-flou). Ces techniques permettent également de doter le système de contrôle d’un degré de rationalité plus important dans la prise de décision. Les algorithmes génétiques peuvent contribuer dans l’amélioration de règles floues.
- Discussion des performances de l’approche de contrôle via d’autres tâches plus complexes (composées de plusieurs tâches élémentaires) tels que le suivi de trajectoires en zigzag, trajectoires circulaires, trajectoires sinusoïdales, etc.
- Test et validation de l’approche de contrôle développée sur un robot virtuel évoluant dans un environnement virtuel (un simulateur graphique.
- Comme dernière perspective, nous comptons appliquer l’approche élaborée sur le manipulateur mobile expérimental RobuTER/ULM.

ANNEXE A

MANIPULATEUR MOBILE ROBUTER/ULM

Principaux repères liés au robot RobuTER/ULM

L'analyse géométrique du manipulateur mobile englobe l'interaction entre la base mobile et le manipulateur. Pour la modélisation du robot, les hypothèses suivantes ont été considérées :

- Il n'y pas de glissement entre les roues motrices de la base mobile et le sol.
- La base mobile ne peut pas se déplacer de côté en raison de la contrainte de non-holonomie.
- Le manipulateur est fixé de façon rigide sur la base mobile.

L'analyse mathématique d'un manipulateur mobile nécessite la définition des principaux repères et transformations matricielles utilisés :

- $R_A = (O_A, \overrightarrow{x_A}, \overrightarrow{y_A}, \overrightarrow{z_A})$: le repère absolu lié à l'environnement.
- $R_B = (O_B, \overrightarrow{x_B}, \overrightarrow{y_B}, \overrightarrow{z_B})$: le repère attaché à la base mobile.
- $R_M = (O_M, \overrightarrow{x_M}, \overrightarrow{y_M}, \overrightarrow{z_M})$: le repère placé à la base du manipulateur.
- $R_E = (O_E, \overrightarrow{x_E}, \overrightarrow{y_E}, \overrightarrow{z_E})$: le repère fixé à l'effecteur.
- ${}^{M}T_E$: la matrice de transformation définissant R_E dans R_M.
- ${}^{B}T_M$: la matrice de transformation qui définit R_M dans R_B.
- ${}^{A}T_B$: la matrice de transformation définissant R_B dans R_A.
- ${}^{A}T_E$: la matrice de transformation qui définit R_E dans R_A.

- (x_m, y_m, z_m) : la position de la base du manipulateur *ULM* dans le repère R_B.

Analyse du modèle géométrique direct du manipulateur ULM

Le MGD d'un manipulateur exprime la relation entre la situation de son effecteur $(x_E, y_E, z_E, \psi_E, \theta_E, \phi_E)$ et les différentes coordonnées articulaires $(Q_1, ..., Q_6)$. Cette situation est calculée dans R_M selon la notation MDH [35]. Dans cette notation, le repère $R_M = (O_M, \overrightarrow{x_M}, \overrightarrow{y_M}, \overrightarrow{z_M})$ se trouve lié à la base du manipulateur. Les autres repères $R_k = (O_k, \overrightarrow{x_k}, \overrightarrow{y_k}, \overrightarrow{z_k})$ sont liés aux différentes articulations et se succèdent jusqu'à atteindre celui de l'effecteur. La notation *MDH* permet de représenter les déplacements de l'articulation *k* relativement à l'articulation *k-1*, grâce à une matrice de transformation d'espace ${}^{k-1}T_k$ donnée par (1) [14] :

$$ {}^{k-1}T_k = \begin{pmatrix} cos\,\theta_k & -sin\,\theta_k & 0 & a_k \\ cos\,\alpha_k * sin\theta_k & cos\,\alpha_k * cos\,\theta_k & -sin\,\theta_k & -d_k * sin\alpha_k \\ sin\,\alpha_k * sin\theta_k & sin\,\alpha_k * cos\theta_k & cos\,\alpha_k & d_k * cos\alpha_k \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1) $$

La matrice de transformation d'espace résultante ${}^{M}T_E$, donnée par (2), illustre mathématiquement la situation de l'effecteur dans le repère de référence R_M tels que ${}^{M}T_2$ définit R_2 dans R_M, ${}^{k-1}T_k$ (k=3..6) définit R_k dans R_{k-1} et ${}^{6}T_E$ définit R_E dans R_6 :

$$ {}^{M}T_E = \begin{pmatrix} t_{11} & t_{12} & t_{13} & x_E \\ t_{21} & t_{22} & t_{23} & y_E \\ t_{31} & t_{32} & t_{33} & z_E \\ 0 & 0 & 0 & 1 \end{pmatrix} = {}^{M}T_2 * {}^{2}T_3 * {}^{3}T_4 * {}^{4}T_5 * {}^{5}T_6 * {}^{6}T_E \quad (2) $$

Chacune des matrices a été calculée selon la représentation *MDH* tels que c_k signifie $cosQ_k$ et s_k signifie $sinQ_k$ (k=1..6) :

$$ {}^{M}T_2 = \begin{pmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{2}T_3 = \begin{pmatrix} c_2 & -s_2 & 0 & 0 \\ 0 & 0 & -1 & -d_2 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{3}T_4 = \begin{pmatrix} 1 & 0 & 0 & a_3 \\ 0 & 0 & 1 & d_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} c_3 & -s_3 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} c_3 & -s_3 & 0 & a_3 \\ s_3 & c_3 & 0 & d_3 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{4}T_5 = \begin{pmatrix} c_4 & -s_4 & 0 & 0 \\ 0 & 0 & -1 & -d_4 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

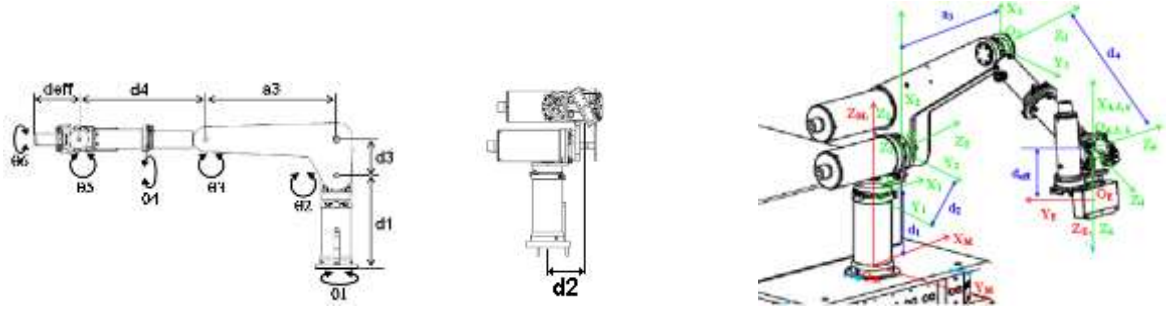
$$ {}^{5}T_6 = \begin{pmatrix} c_5 & -s_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_5 & c_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

$$ {}^{6}T_E = \begin{pmatrix} c_6 & -s_6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s_6 & c_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_{eff} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} c_6 & -s_6 & 0 & 0 \\ 0 & 0 & -1 & -d_{eff} \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} $$

(x_E, y_E, z_E) sont les positions cartésiennes de l'effecteur selon les axes $\overrightarrow{x_M}$, $\overrightarrow{y_M}$ et $\overrightarrow{z_M}$. Pour les rotations, la représentation non-redondante utilisée est celles des angles d'Euler classiques $(\psi_E, \theta_E, \phi_E)$ selon les trois axes précédents [7]. Le calcul de ces orientations est donné par (3) :

$$\begin{aligned} \psi_E &= \begin{cases} Atan2(-t_{13},\ t_{23}) \\ Atan2(-t_{13},\ t_{23}) = \psi_E + 180° \end{cases} \\ \theta_E &= Atan2\left(sin\ \psi_E * t_{13} - cos\ \psi_E * t_{23},\ t_{33}\right) \\ \phi_E &= Atan2(-cos\psi_E * t_{12} - sin\ \psi_E * t_{22},\ cos\ \psi_E * t_{11} + sin\ \psi_E * t_{21}) \end{aligned} \tag{3}$$

Les différents paramètres *MDH* du manipulateur *ULM* α_k, d_k, θ_k et a_k sont donnés par Fig. 4 [29].



Paramètres *MDH* et principaux repères liés au manipulateur *ULM*

Les valeurs de ces paramètres ainsi que les limites articulaires du manipulateur *ULM* sont données par Tab. 1 :

K	α_k (°)	d_k (mm)	θ_k	a_k(mm)	Q_{Min}(°)	Q_{Max}(°)
1	0	d_1=290	θ_1	0	-95	96
2	90	d_2=108.49	θ_2	0	-24	88
3	-90	d_3=113	0	a_3=402	–	–
4	90	0	θ_3	0	-2	160
5	90	d_4=389	θ_4	0	-50	107
6	-90	0	θ_5	0	-73	40
7	90	d_{eff}=220	θ_6	0	-91	91

Paramètres *MDH* et limites articulaires du manipulateur *ULM*

ANNEXE B STRUCTURE DU FICHIER *TRACE*

Scénario 1 (Jointfootstep = 1, Base translation footstep= 5, Base rotation footstep = 1)

Configuration_Init(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)

Effector_Init(0432.00, -0108.49, 0434.00) ----> Effector_Fin(-330.0, -630.0, 1080.0)

N° Itération	Q1(°)	Q2(°)	Q3(°)	Q4(°)	Q5(°)	Q6(°)	X_Base	Y_Base(	Theta_	X_E	Y_E	Z_E	Error (mm)
0	0	0	0	0	0	0	0	0	0	432	-108,49	434	1126,9129
1	0	0	0	0	0	0	0	0	-1	142,1194	-422,1329	434	826,6931
2	0	0	0	0	0	0	0	0	-2	-278,4251	-347,6687	434	706,8854
3	0	1	0	0	0	0	0	0	-2	-282,002	-355,4843	441,0914	697,0415
4	0	2	0	0	0	0	0	0	-2	-285,5268	-363,1861	448,3317	687,1479
5	0	3	0	0	0	0	0	0	-2	-288,9984	-370,7719	455,7188	677,2055
6	0	4	0	0	0	0	0	0	-2	-292,416	-378,2393	463,2503	667,2152
7	0	5	0	0	0	0	0	0	-2	-295,7782	-385,586	470,924	657,178
8	0	6	0	0	0	0	0	0	-2	-299,0843	-392,8098	478,7376	647,0946
9	0	7	0	0	0	0	0	0	-2	-302,3331	-399,9085	486,6886	636,9662
10	0	8	0	0	0	0	0	0	-2	-305,5236	-406,88	494,7746	626,7937
65	2	56	5	0	0	0	0	0	-2	-367,6383	-598,4269	1031,2128	69,2365
66	3	56	5	0	0	0	0	0	-2	-357,6163	-604,5382	1031,2128	61,5723
67	3	57	5	0	0	0	0	0	-2	-356,9291	-602,8018	1042,7816	53,3866
68	3	57	6	0	0	0	0	0	-2	-358,7351	-607,3653	1052,2096	45,9385
69	4	57	6	0	0	0	0	0	-2	-348,5585	-613,3199	1052,2096	37,3491
70	4	57	7	0	0	0	0	0	-2	-350,2265	-617,7595	1061,7218	29,8837
71	5	57	7	0	0	0	0	0	-2	-339,8698	-623,5639	1061,7218	21,7469
72	5	58	7	0	0	0	0	0	-2	-339,0646	-621,3018	1073,4217	14,181
73	6	58	7	0	0	0	0	0	-2	-328,6478	-626,9109	1073,4217	7,3922
74	6	58	8	0	0	0	0	0	-2	-330,053	-631,0887	1083,0936	3,28
75	6	58	8	0	-1	0	0	0	-2	-329,5454	-629,5795	1079,5996	0,7374

RÉFÉRENCES BIBLIOGRAPHIQUES

1. A. Agah , K. Tanie. Fuzzy logic controller design utilizing multiple contending software agents. *Fuzzy Sets and Systems, 106(2)*, pp.121-130, 1999.
2. Akli I., Haddad M., Bouzouia B. & Achour N., "Trajectory Generation for Operational Task Execution with Manipulability Analysis". *The International Conference on Advanced Robotics (ICAR2011)*, Tallinn University of Technology, Estonia, 20-23 June 2011.
3. Akli I., "Elaboration d'une stratégie de coordination de mouvements pour un manipulateur mobile redondant", *Mémoire de magister, USTHB*, Juillet 2007.
4. R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert, "Multi-robot cooperation in the Martha project", *IEEE Robotics and Automation Magazine, 5(1)*, pp. 36-47, 1998.
5. Albus J. S., "Hierarchical Interaction between Sensory Processing and World Modeling in Intelligent Systems". *The International Symposium on Intelligent Control (ISIC)*, Penn Tower Hotel, Philadelphia, Pennsylvania, USA, pp. 866–875, 5-7 September 1990.
6. Alfaro C., Ribeiro M. I. & Lima P., "Smooth Local Path Planning for a Mobile Manipulator". *Robotica 2004, The 4th Portuguese Robotics Festival*, Protugal, pp. 127-134, 2004.
7. Bayle B., Fourquet J.-Y., Renaud M., "From Manipulation to Wheeled Mobile Manipulation: Analogies and Differences", *The International IFAC Symposium on Robot Control (Syroco2003)*, pp. 97-104, 2003, Wroclaw, Poland.

8. Beer M., d'Inverno M., Luck M., Jennings N., Preist C., Schroeder M., "Negotiation in Multi-Agent Systems", *Workshop of the UK Special Interest Group on Multi-Agent Systems (UKMAS'98)*, 1998.
9. R. A. Brooks, "A robust layered control system for a mobile robot", *IEEE Journal of Robotics and Automation*, pp. 14-23, 1986.
10. Caselli S.,Fantini E.,Monica F.,Occhi P. & Reggiani M., "Toward a Mobile Manipulator Service Robot for Human Assistance". *Robocare Workshop*, Italy, 2003.
11. B. Chaib-Draa., I. Jarras, B. Moulin, "Systèmes multi-agents : principes généraux et applications", *Principes et architectures des systèmes multiagents, J.-P. Briot et Y.Demazeau (Eds.), Hermès*, 2001.
12. Dean T. & Wellman M., "Planning and Control", *Morgan-Kaufmann*, 1991.
13. S. Delarue, Ph. Hoppenot, E. Colle, "A multi Agent controller for a mobile arm manipulator", *ICINCO 2007*.
14. Dombre E. & Khalil W., "Modeling, Performance Analysis and Control of Robot Manipulators". *Control Systems, Robotics and Manufacturing Series, Hermes Science Publishing*, 2007.
15. D. Duhaut, "Distributed Algorithm for High Control Robotics Structures", *Int. Conf. on Artificial Intelligence*, 1999, Vol.1, pp 45-50.
16. V. Dupourqué. "Les contrôleurs de robots". *Collection Novotique AFRI*, Paris, 1987.
17. Feil-Seifer D. & Mataric M. J., "B^{3}IA: A control architecture for autonomous robot-assisted behavior intervention for children with Autism Spectrum Disorders". *The 17th International Symposium on Robot and Human Interactive Communication (ROMAN08)*, Technische Universität München, Germany, 2008.
18. J. Ferber, "Les systèmes Multi agents Vers une intelligence collective", IIA Informatique Intelligence Artificielle InterEditions – septembre 1995
19. Ferraz de Camargo R., "Architecture matérielle et logicielle pour le contrôle d'exécution d'un robot mobile autonome". *Thèse de doctorat en Robotique, Université de Toulouse III*, Toulouse, France, 1991.
20. http://www.fipa.org
21. S. Fleury, « Architecture de contrôle distribuée pour robots mobiles autonomes : principes, conception et applications », LAAS, 1996.
22. Floroian D., Moldoveanu F., "Using Robosmith for Multi-agent Robotic System", *Bulletin of the Transilvania University of Brasov, Series I: Engineering Sciences, 3(52)*, 2010.

23. G. Foulon, “Génération de mouvements Coordonnés pour un ensemble constitué d’une plateforme mobile à roues et d’un bras manipulateur“, *Thèse de Doctorat, LAAS-CNRS, Institut National des Sciences Appliquées*, Toulouse, France, 1998.

24. Gorla B. & Renault M., “Modèles des robots manipulateurs : application à leurs Commandes“, *Editions Cepadues*, 1984.

25. Z. Guessoum, “A Hybrid Agent Model: a Reactive and Cognitive Behavior”, *IEEE ISAD97*, Berlin, Germany, pp. 25-32, April 1997.

26. Harrigan R. W. & Bennett Ph. C., “Dexterous Manipulation: Making Remote Manipulators Easy to Use”, Unlimited Release, November 2001.

27. Hentout A., Messous M. A., Oukid S., & Bouzouia B., “Multi-Agent Fuzzy-Based Control Architecture for Autonomous Mobile Manipulators: Traditional Approaches and Multi-Agent Fuzzy-Based Approaches”. *In Intelligent Robotics and Applications, Lecture Notes in Computer Science (LNCS), Volume 8102*, 2013, pp. 679-692.

28. Hentout A., Bouzouia B., Akli I., Toumi R., “Mobile Manipulation: A Case Study”. *Robot Manipulators, New Achievements, Aleksandar Lazinica and Hiroyuki Kawai (Ed.)*, pp. 145-167, 2010.

29. Hentout A., “Architecture Multi-agents Dédiée au Contrôle des Manipulateurs Mobiles : Application au Robot RobuTER/ULM ”, *Thèse de doctorat, USTHB*, Février2012.

30. Huang S., Aertbeliën E. & Van Brussel H., “An Autonomic

31. Huntsberger T. L., Pirjanian P., Trebi-Ollennu A., Das Nayar H., Aghazarian H., Ganino A. J., Garrett M., Joshi S. S. & Schenker P. S., “CAMPOUT: A Control Architecture for Tightly Coupled Coordination of Multi-robot Systems for Planetary Surface Exploration”. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, 33(5)*, 2003.

32. Iñigo-Blasco P., Diaz-del-Rio F., Romero-Ternero C., Cagigas-Muñiz D., Vicente-Diaz S., “Robotics software frameworks for multi-agent robotic systems development”, *Robotics and Autonomous Systems, 60(1)*, pp. 803-821, 2012.

33. http://33.tilab.com/

34. Jennings N., Sycara K., Wooldridge M., "A Roadmap of Agent Research and Development", *Autonomous Agents and Multi-Agent Systems, 1(1)*, pp. 7 - 38, July 1998.

35. Khalil W. & Kleinfinger J. F., “A new geometric notation for open and closed-loop robots”. *The International Conference on Robotics and Automation (ICRA’86)*, San Francisco, USA, pp. 1175-1180, 1986.

36. Y.Kunii, Y.Kuroda,T.Kubota, “ Development of micro-manipulator for tele science by lunar rover: Micro5”, *Elsevier trans. Acta Astronautica, 52(1)*, pp. 433–439, 2003.

37. T. Laengle, T. Hoeniger, U. Rembold, H.Woern, “Cooperation in Human-Robot-Teams”, *Int. Conf. Informatics and Control (ICI&C'97),* St. Petersburg, Russie, Juin 9-13, 1997.

38. P. Lucidarme and A. Liégeois, "Apprentissage de comportements réactifs pour des ensembles de robots mobiles", *JJCR17*, Paris, pp. 33-39, 2003.

39. Luis Fernando Carroll Janer, Vers la maîtrise du développement d’un contrôleur temps réel sûr de fonctionnement pour les robots manipulateurs. *Thèse de doctorat en Automatique et Informatique Industrielle, Institut National des Sciences Appliquées de Toulouse*, Octobre 1999.

40. Lussier B., "Tolérance aux fautes dans les systèmes autonomes". *Thèse de doctorat en Systèmes Informatiques, Institut National Polytechnique de Toulouse*, École Doctorale Systèmes, France, avril 2007.

41. Mataric M. J., “Behavior-Based Control: Main Properties and Implications”. The Workshop on Intelligent Control Systems, *International Conference on Robotics and Automation (ICRA’92)*, Nice, France, May 1992.

42. Mataric M. J., “Behavior-Based Robotics as a Tool for Synthesis of Artificial Behavior and Analysis of Natural Behavior”. *Trends in Cognitive Science, 2(3)*, pp. 82–87, 1998.

43. McBureny, P et al., “AgentLink III : A Coordination Network for Agent-Based Computing”, http://dbs.cordis.lu/fepcgi/srchidadb?ACTION=D&CALLER=PROJ_IST&QM_EP_RCN_A=71184; December 14, 2004.

44. Messous M. A., Hentout A., Oukid S., Bouzouia B., “Developing a Multi-agent Fuzzy-based Control Architecture for Autonomous Mobile Manipulators”, *The 10th International Conference on Informatics, Automation and Robotics (ICINCO’13), Doctoral Consortium*, pp. 15-22, 29 - 31 July 2013, Reykjavik, Iceland.

45. Nagatani K., Hirayama T., Gofuku A., Tanaka Y., “Motion planning for mobile manipulator with keeping manipulability”. *The International conference on Intelligent Robots and Systems (IROS2002),* Switzerland, 2002.

46. Nana L., “Architectures Logicielles pour la Robotique”. *Journées Nationales de Recherches en Robotique (JNRR'05)*, Guidel, Morbihan, France, 5-7 octobre 2005.

47. Nebot R. P., “Agent-based Architecture for Multi-robot Cooperative Tasks: Design and Applications”. *Ph. D. thesis in Informatics*, University Jaume I, Spain, 2007.

48. Nikoobin A., Rahimi H. N., “Analyzing the Wheeled Mobile Manipulators with Considering the Kinematics and Dynamics of the Wheels”, *International Journal of Recent Trends in Engineering, 1(5)*, pp. 90- 92, 2009.

49. R. T. Pack, “IMA: The Intelligent Machine Architecture”, *Ph.D. Thesis, Université de Vanderbilt*, Nashville, Tennessee, USA, 2003.

50. Russell S. J., Norvig P., "Artificial Intelligence. A Modern Approach", *Prentice-Hall, Englewood Cliffs*, 1995.

51. Sugar T. & Kumar V., “Decentralized Control of Cooperating Mobile Manipulators”. *The International Conference on Robotics and Automation (ICRA’98)*, Leuven, Belgium, pp. 2916-2921, 1998.

52. P. Tournassoud, "Planification et contrôle en robotique Application aux robots mobiles et manipulateurs", *Edition Hermes Traité des nouvelles Technologies*, série robotique, 1992.

53. V. Padois, "Enchaînements dynamiques de tâches pour des manipulateurs mobiles à roues". *Thèse de Doctorat de l’Institut National Polytechnique de Toulouse*, 2005.

54. Weiss G., "A Modern Approach to Distributed Artificial Intelligence", *The MIT Press, Cambridge*, Massachusetts, 1999.

55. Yu Z., Yan J., Zhao J., Gao Y. & Cai H., “Architecture of Multi-Agent-Based Multi-Operator Multi-Mobile-Manipulator Teleoperation System”. *C. Xiong et al. (Eds.): ICIRA 2008, Part II, LNAI 5315, Springer-Verlag Berlin Heidelberg*, pp. 17–26, 2008.

56. S. Huang, E. Aertbelien, H. Van Brussel, “Behavior-Based Software Architecture for Mobile Manipulator”. The International *Conference on Advanced Robotics and its Social Impacts*, Taipei, Taiwan, 23-25 August 2008.

57. Colle E., Nait Chabane K., Delarue S. & Hoppenot P., “ARPH : Comparaison d'une méthode classique et d'une méthode utilisant la coopération homme-machine pour exploiter la redondance de l'assistant robotisé”. 4ième Conférence HANDICAP2006 «Nouvelles Technologies au service de l’homme», 7-9 juin 2006, Paris, France.

58. Azevedo J. L., Bernardo Cunha M. & Almeida L., “Hierarchical Distributed Architectures for Autonomous Mobile Robots: a Case Study”. The Conference on Emerging Technologies and Factory Automation (ETFA2007), pp. 973-980, Patras, Greece, 25-28 September 2007.

59. Erden M. S., Leblebicioglu K. & Halici U., “Multi-agent System-Based Fuzzy Controller Design with Genetic Tuning for a Mobile Manipulator Robot in the Hand Over Task”. Journal of Intelligent and Robotic Systems, 39(3), pp. 287-306, 2004.

60. X.J. Wu, Q. Li and K.H. Heng, “Development of a general manipulator path planner using fuzzy reasoning”, Industrial Robot: An International Journal, 32/3 (2005) 248–258.