R e m e r c i e m e n t s

Je voudrais d'abord remercier monsieur Le professeur Alain Liégeois, pour m'avoir accueilli au sein de son équipe.

Monsieur Philippe Rongier, qui a toujours été disponible pour répondre à mes nombreuses questions.

Monsieur Dauchez, responsable du département robotique.

Monsieur Geovany Borges pour ses précieux conseils en électronique.

Et enfin l'ensemble des thésards et permanents du département robotique du L.I.R.M.M.

Table des matières

Introduction Générale

Les systèmes multi-agents se sont de plus en plus développés ces dernières années dans différents domaines. On constate de manière générale, une décentralisation des systèmes de commande. L'architecture traditionnelle, consistait, il y a encore quelques années à n'avoir qu'un seul et unique superviseur qui contrôlait l'ensemble du processus. Aujourd'hui on s'aperçoit qu'il vaut mieux, pour certaines applications, avoir plusieurs processus simples qui évoluent en parallèle. Cela présente surtout l'avantage d'être très fiable, en effet, même si une partie du processus échoue, la redondance permettra tout de même de finir la mission.

En robotique mobile, nous constatons que les applications classiques se présentent généralement de la forme suivante : les robots doivent atteindre une cible dans l'environnement, pour la déplacer vers un autre endroit ou y effectuer une tâche, parfois dans un environnement inconnu. Pour ce type d'applications, les systèmes multi-agents sont parfaitement adaptés. La robotique mobile cognitive propose des solutions, comme par exemple la création d'une carte des lieux. Mais ces solutions sont généralement gourmandes en temps de calcul et une panne paralyse l'ensemble du système. De plus les approches multi-agents proposent souvent l'avantage de n'utiliser que des robots réactifs, peu coûteux et beaucoup plus simples à fabriquer.

De nombreuses théories multi-agents sont basées sur la marche aléatoire, cela signifie que l'on ne peut pas connaître la trajectoire des robots à l'avance. D'ailleurs à chaque simulation, les agents se déplacent suivant un nouveau parcours. Cela pose tout de même des problèmes, car il est très dur d'en estimer les performances. Une des méthodes employées pour estimer la fiabilité des systèmes est de réaliser un nombre suffisamment élevé de simulations pour espérer couvrir l'ensemble des cas de façon représentative. Malheureusement cela demande beaucoup de temps, et peut monopoliser une machine plusieurs heures, voir plusieurs jours. Alors, pour obtenir des résultats rapidement, M. Alain Liegeois et Philippe Rongier ont proposé une modélisation des systèmes grâce aux chaînes de Markov. La méthode décrite en [1] permet de modéliser des tâches du type fourragement.

Dans cette application, la seule coopération existante est en réalité le fait que chaque agent ai le même objectif. Mais la coopération ne prends pas toute son ampleur dans le sens ou un agent n'a pas nécessairement besoin d'un autre pour finir sa mission. Alors il semblait intéressant de s'occuper de tâches purement coopératives. C'est pour cela qu'il a été a proposé l'exemple de l'aveugle et du paralytique faisant référence à une fable de Jean Pierre Claris de Florian (1755-1794) [2]. Ce modèle sera décrit de façon plus complète dans le premier chapitre de ce document.

Nous développerons dans ce document une rapide présentation des travaux antérieurs dans le domaine, puis une présentation de l'application qui nous intéresse ici. Ensuite nous reviendrons sur les différentes approches : par chaîne de Markov, en simulation puis sur le système réel. Enfin nous analyserons les résultats et conclurons sur la fiabilité de nos théories.

P r é s e n t a t i o n G é n é r a l e

2.1. Introduction

De nombreux travaux ont déjà été menés dans le domaine du multi-agents. Ces travaux ont été réalisés aussi bien par des informaticiens que par des roboticiens. Cela prouve bien que c'est un concept qui touche de nombreux domaines et que ses applications sont vastes. De plus en plus, les méthodes employées se complexifient et il n'est pas rare de voir s'ajouter des fonctions d'apprentissage. Toutes ces méthodes deviennent de plus en plus longues à simuler et il est parfois difficile de conclure quand à leur efficacité.

2.2. Etat de l'art

2.2.1. Les systèmes multi-agents

Les premières études sur la coopération multi-agents avaient pour but l'étude du milieu animal, notamment celui des insectes. Ces études ont rapidement montrée que chaque insecte n'avait aucune notion de son rôle dans la tâche globale représentée par l'ensemble de l'essaim. Ses actions étaient uniquement guidées par des changements de comportement. Ce phénomène a rapidement intéressé les roboticiens qui se sont inspirés de ce milieu afin de réaliser des flottes de robots coopérants. Les premiers articles concernant les systèmes multi-agents remontent au début des années 80 pour la robotique. Les premiers objectifs furent de définir des méthodes simples et fiables afin de réaliser des tâches courantes [3,4]. Ces tâches, généralement inspirées du milieu animal (principalement les fourmis), peuvent se décomposer en trois grandes familles de tâches génériques:

- La mine : consiste à trouver une, ou plusieurs, mine(s) et à en rapporter le minerai à la base. C'est une des applications les plus courantes. Les principaux intérêts de cette tâche résident dans le fait que l'environnement n'est pas connu à l'avance, que les robots sont obligés de faire des aller retours entre la base et la source. Et ce sont ces points en particulier qui font que la coopération prend parfois toute son ampleur.

- La consommation : similaire à la précédente, la consommation est une tâche qui se différencie de la mine dans le fait que l'action est effectuée sur place. Dans la mine, le minerai est ramassé, puis ramené à la base alors que pour la consommation, on réalise une tâche sur place comme par exemple de la peinture ou de l'assemblage. C'est parfois cette action qui va nécessiter la coopération entre les robots.
- L'exploration : consiste à parcourir la plus grande surface possible de l'environnement. Ici aussi il est sûr que, dans bien des cas, plus les agents sont nombreux, plus la tâche sera réalisée rapidement. C'est justement sur ce point que la coopération devient importantes, le système global doit être bien pensé pour éviter de voir un agent explorer une zone qui l'a déjà été. C'est également pour cela que la communication est l'un des éléments essentiels des systèmes multi-agents.

La communication a été très tôt un des atouts principaux des systèmes multi-agents. Il existe différents types de communications. Une des communications les plus simples est basée sur l'exemple du petit poucet [3]. Elle consiste à déposer sur le terrain des marques qui pourront être captées ou ramassées par d'autres agents. Ces marques peuvent servir à définir un territoire qui a déjà été visité ou encore le chemin de la mine. On appelle ce type de communication indirecte. Mais il existe d'autres types de communications : directs. Ces communications servent à transmettre des informations sur sa position, son environnement proche ou encore son état interne. Par exemple, dans le modèle basé sur la satisfaction [5] l'un des éléments principaux de la communication est la transmission d'un vecteur qui représente la satisfaction de l'agent.

Les différents principes qui viennent d'être brièvement présentés sont les bases de l'approche multi-agents. La majeure partie des travaux qui ont été réalisés ces dernières années ont permis d'apporter des compléments aux modèles, permettant ainsi de parfaire leurs performances. On constate également que les systèmes multi-agents sont fréquemment associés à l'intelligence artificielle. Notamment les travaux de Rodney Brooks, sur la gestion par apprentissage d'un robot humanoïde basée sur les théories multi-agents [6].

Ces différentes méthodes, de plus en plus complexes doivent être modélisées avant implémentation afin d'en estimer leurs performances. Nous allons voir, dans la partie qui suit deux modélisations courantes, soit par réseaux de Petri stochastiques, soit par chaînes de Markov.

2.2.2. Les modélisations

L'article [1] présente un modèle basé sur les réseaux de Petri stochastiques. L'idée est de modéliser une tâche du type la mine grâce à un réseau de Petri stochastique. Le nombre d'échantillons est fini, les deux mines que comporte le terrain sont donc épuisables. Les agents sont capable d'adopter les comportements basiques suivants :

- Remonter le gradient afin d'atteindre la base (remonter un signal jusqu'à sa source)
- Ramasser et déposer des échantillons ou des marques
- Se déplacer de façon aléatoire

La mission globale et le comportement de chaque agent ont été modélisé grâce à des réseaux de Petri stochastiques (respectivement figure1.a et 1 .b).

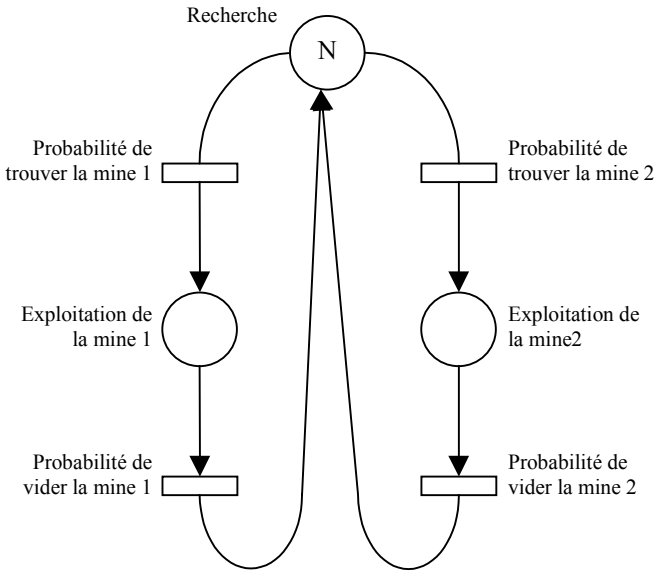

figure1.a : Représentation de haut niveau de la mission globale

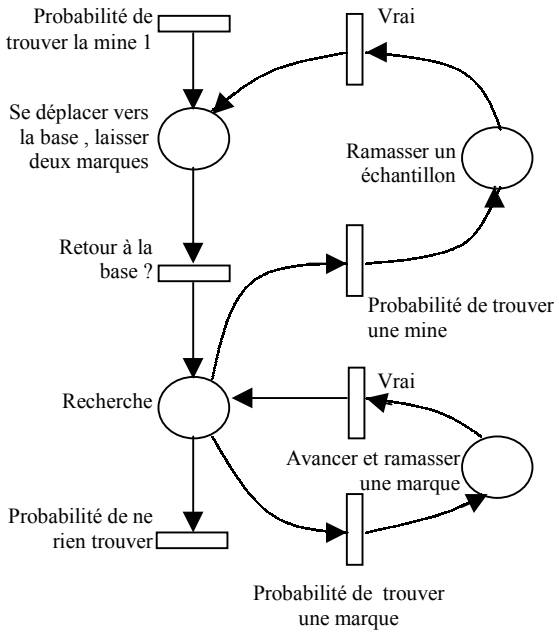

Figure 1.b : Représentation du comportement d'un robot

Ce type de représentation permet de définir le fonctionnement du système d'un point de vue global. On constate que chaque transition est une probabilité et chaque place représente un des comportements définis précédemment. Ensuite les auteurs créent la chaîne de Markov associée à ces réseaux de Petri et en déduisent l'évolution du système en fonction du temps.

D'après leurs études, nous pouvons en conclure que chacune des trois tâches génériques (présentées en 2.2.1.) pourra aisément se modéliser de la même façon. Il était

donc intéressant d'étudier des systèmes plus complexes afin de valider la généralisation de la méthode.

A.Martinoli, A.J. Ijspeert et F. Mondada ont récemment proposé [7] l'étude d'un système qui se démarque des 3 tâches génériques présentées précédemment. Le principe de l'application est inspiré des termites qui ramassent des brindilles, dans le but de les regrouper en un ou plusieurs tas. Comme dans les exemples précédents, le comportement d'une termite peut s'assimiler à celui d'un agent réactif. Le principe générale de chaque agent est le suivant : il se déplace de façon aléatoire dans l'environnement, dès qu'il détecte une brindille, il la ramasse. Il repart dans une phase de recherche tout en transportant la brindille. Dés qu'il rencontre à nouveau une brindille, il la dépose à côté. Cet exemple est tout à fait révélateur de la puissance des systèmes multi-agents car à la fin d'un certain temps on s'aperçoit que toutes les brindilles ont été rassemblées en quelques tas, voir même un seul. Les agents ne réagissent pas de façon cognitive, ils exécutent une tâche basique à partir de fonctions primitives (se déplacer, ramasser, déposer…), les agents ne cherchent même pas à déposer la brindille près du plus gros tas. C'est la mission globale qui va permettre d'arriver au but voulu : émergence d'un comportement.

Leur étude se décompose en trois phases. Leur première approche consiste à simuler le système grâce à une plate-forme de développement : Webots. [8]. Ensuite, le système est modélisé grâce à une chaîne de Markov, ce qui permet d'obtenir des résultats de simulation plus rapidement. Et enfin, un des points forts de cet article reste sans doute la mise en application de leurs expériences grâce des robots mobiles du type Képhéra. Une partie de cet article est dédiée aux innovations technologiques mis en œuvre pour la réalisation de cette manipulation et aux différences entre la simulation et la réalité [9]. Malgré toutes ces difficultés, ils prouvent qu'il est désormais possible de réaliser des applications des systèmes multi-agents. Et de surcroît que ces applications sont parfaitement fidèles à leurs modèles, qu'ils soit simulés ou modélisés par chaînes de Markov puisque leurs résultats concordent parfaitement.

Dans les différents articles que nous venons de présenter, la coopération entre les agents n'est pas un des atouts mis en avant. En effet, lorsque deux agents coopèrent, cela ne permet pas d'augmenter leurs performances propres mais uniquement les résultats du système global. Nous souhaitions valider les résultats présentés dans [1] sur un système innovateur ou l'entre aide sociale permettrait d'augmenter de façon significative les performances des agents coopérants.

2.3. Présentation de notre système : l'aveugle et le paralytique

Le système que nous avons choisi d'étudier est inspiré d'une fable de Jean Pierre Claris de Florian (1755-1794) [2] : L'aveugle et le paralytique. Cette fable raconte comment un aveugle et un paralytique en sont arrivés à s'entraider afin de réduire leurs malheurs. Le paralytique guide l'aveugle, qui en échange le porte. Nous aurons donc deux types d'agents : des aveugles et des paralytiques. L'idée générale du système est basée sur la coopération entre ces deux agents. Leur but est d'atteindre des cibles (dans notre cas des restaurants). Lorsqu'un aveugle entend un paralytique, il le prend sur son dos, et le paralytique, qui peut voir les restaurants, le guide. En revanche, si un aveugle trouve par hasard le restaurant sans l'aide d'un paralytique, il y va seul.

Pour les besoins de la simulation, nous considérerons que le terrain est discrétisé. Il a une taille de 60 x 40 cases. Les positions initiales des aveugles et des paralytiques sont aléatoires, celles des restaurants sont fixes. Les aveugles peuvent détecter le bruit émis par les restaurants ou par les paralytiques. Dans nos exemples, les paralytiques peuvent être entendus à condition de se trouver à moins de 5 cases de celui-ci. Cette partie du terrain est appelée zone d'appel. Les restaurants ont une zone d'appel de 1 case. Un paralytique peut voir le restaurant à une distance de 15 cases (figure 2).

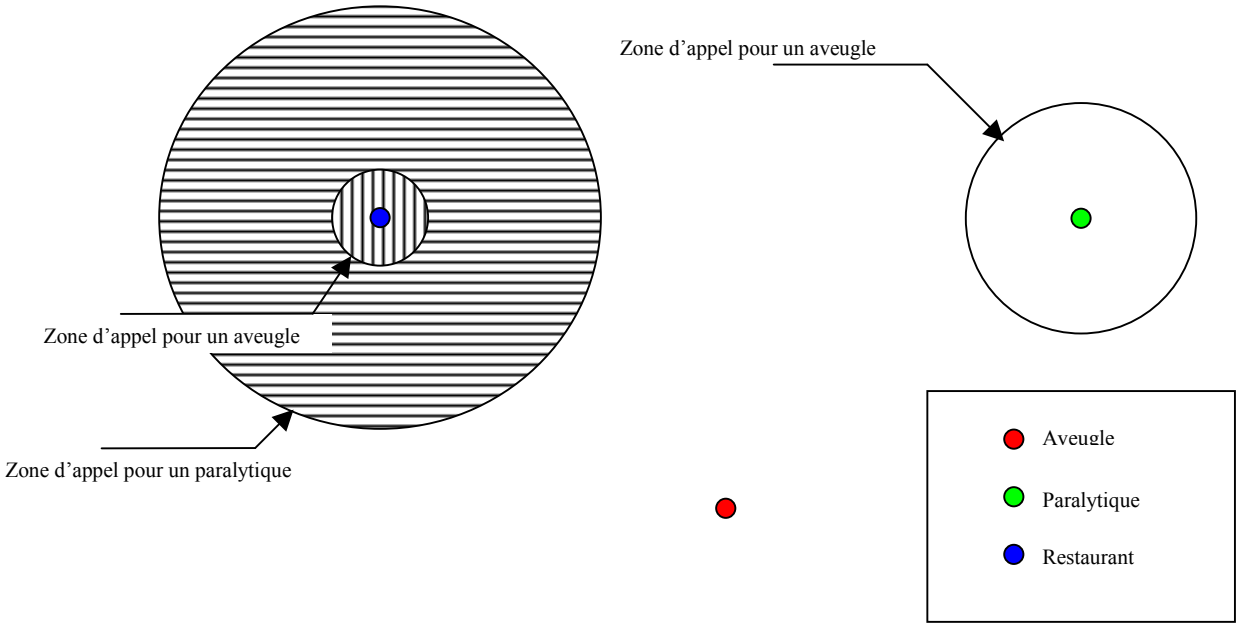

figure 2 : Exemple de zones d'appel pour un paralytique et pour un restaurant. (les échelles ne sont pas respectées)

Dans notre étude, l'environnement ne comporte pas d'obstacles, les agents évoluent librement. Les seules zones pouvant être considérées comme telles sont les bords du terrain.

La figure 2 schématise le fonctionnement de ces zones d'appel. Lorsqu'un aveugle seul rentre dans la zone hachurée verticale (appel à 1), il est capable de localiser la position du restaurant et de s'y diriger. Lorsqu'il porte un paralytique, il peut localiser le restaurant à partir de la zone qui est hachurée de façon horizontale (appel à 15), car le paralytique a la possibilité de voir le restaurant de plus loin. De même, dés qu'un aveugle rentre dans la zone d'un paralytique, il est capable de le localiser pour venir le chercher. Il est donc évident, que trouver un paralytique est un atout considérable pour l'aveugle, qui pourra trouver le restaurant plus facilement. Nous noterons toutefois, qu'un aveugle ne peut porter qu'un seul paralytique à la fois.

Un autre point qui démarque cette application des travaux antérieurs est le type de déplacement aléatoires. A.Martinoli ne précise pas dans son article comment sont effectués les déplacements, mais il restreint son étude en émettant l'hypothèse que la probabilité d'atteindre une zone du terrain est proportionnelle à sa surface. Nous verrons plus tard que ce type d'hypothèse n'est pas valable pour toutes les marches aléatoires, par exemple celle que nous allons utiliser ici : la connexité 8. Comme nous considérons que le terrain est discrétisé, les agents se trouvent toujours sur un case, et pour connaître le déplacement, l'agent tire une des 8 cases voisines au hasard (de façon équiprobable) puis s'y déplace (figure 3).

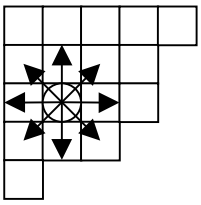

figure 3 : Les 8 déplacement possibles d'un aveugle à chaque cycle.

Après avoir brièvement décrit les travaux antérieurs et le système que nous allons étudier, nous allons maintenant préciser nos objectifs, et les différents moyens d'étude. La première approche, sera celle de la simulation. Rapide à mettre en œuvre , facilement configurable, elle possède toutefois l'inconvénient d'être coûteuse en temps de calcul et exige de nombreuses heures pour obtenir un résultat reflétant l'ensemble des cas possibles. C'est pour cela que nous allons présenter une autre approche basée sur les chaînes de Markov. Cette méthode devrait nous donner les mêmes résultats que la simulation de façon nettement plus rapide. Et enfin, avant de comparer l'ensemble des performances, une implémentation sur système réel nous permettra de confirmer de la fiabilité et de la validité de ces recherches. Mais, détaillons maintenant les différents modèles que nous allons utiliser.

Différentes Approches

3.1 Introduction

Maintenant que le système est présenté, les différentes méthodes qui vont nous permettre de le modéliser vont être décrites. Tout d'abords, nous présenterons les deux simulateurs qui permettront de valider les résultats théoriques et détaillerons les fonctionnalités de chacun, et ce qui a motivé notre choix.

Ensuite, nous reviendrons sur la modélisation par chaîne de Markov. Dans un premier temps, la modélisation graphique de ce système sera détaillée. Nous décrirons également comment, grâce à quelques résultats de simulations, l'ensemble des paramètres de notre chaîne de Markov a pu être déterminé. Cela va permettre d'obtenir des résultats très rapide quelques soit le nombre d'aveugles ou de paralytiques.

Et enfin, le système réel sera présenté : nous avons créé quelques robots réactifs. Les choix technologiques qui nous ont permis de réaliser ce système à moindre coût seront exposés. Ensuite, nous décrirons les conditions de la manipulation qui devraient permettre de confirmer les résultats estimés par le simulateur et la modélisation Markovienne.

3.2 Les simulateurs

Pour pouvoir étalonner notre système par chaîne de Markov, il est nécessaire de réaliser quelques « simulations étalons », mais nous reviendrons sur ce principe dans le paragraphe suivant. Ensuite, il doit permettre de vérifier que notre modèle probabiliste est fiable et correspond bien aux résultats de simulation. Ces dernières doivent permettre de nous donner chacun des paramètres représentatifs de la simulation : nombre de paralytiques découverts, durée de la recherche d'un aveugle etc ...

Nous disposons désormais de deux simulateurs : un simulateur développé sous MadKit (Multi-Agents Développement Kit) et un second simulateur programmé durant ce stage sous MatLab.

3.2.1 Le simulateur MadKit

D'abord revenons sur MadKit qui est une plate-forme de développement dédiée principalement à la simulation des systèmes multi-agents [10]. MadKit a été développé au LIRMM par Jacques Ferber et Olivier Gutknecht sous Java 1.1. Cette plate-forme présente essentiellement l'avantage de posséder une programmation orientée objet, et une approche Agent-Groupe-Rôle. Cela permet de structurer les systèmes avec une vision qui s'approche du concept multi-agents.

Malheureusement, MadKit a été développé sous Java 1.1 et fonctionne grâce à la machine virtuelle. Cela permet principalement une portabilité des logiciels, mais accuse une augmentation des temps de calculs surtout sur les stations NT dont nous disposons. Ce défaut est accentué par l'utilisation de processus parallèles (thread) , ce qui est souvent inévitable dans la programmation des systèmes multi-agents.

3.2.2. Le simulateur Matlab

Lorsque nous réalisons des simulations, notre but principal est de comparer ces résultats avec le modèle par chaîne de Markov. Nous devons donc, pour avoir des résultats représentatifs de l'ensemble du système, effectuer un grand nombre de simulations. (entre mille et un million). L'utilisation du simulateur sous MadKit demanderait plusieurs jours pour nous donner ces résultats. Nous en avons donc développé un nouveau sous MatLab. Ce dernier est un langage interprété, ce qui lui vaut la réputation d'être gourmand en temps de calcul sur des instructions comme les boucles ou les conditions. En revanche nous observons une grande rapidité lors des calculs matriciels. Il ne faut pas oublier qu'après la simulation il y a une mise en forme des résultats, et c'est spécialement sur ce point que MatLab montre son efficacité sur MadKit. Toutefois, une simulation de notre système sous Matlab est deux à trois fois plus rapide que sous MadKit.

Il est évident que c'est la simplicité de notre système qui a permis la programmation sous MatLab. Si l'on désirait implémenter un système multi-agents plus complexe, le problème deviendrait vite insurmontable. En effet MatLab ne permet pas l'exécution de processus parallèles. Heureusement, nous travaillons dans un environnement discret, tant pour le terrain que pour le temps. Nous comptons le temps en nombre de cycles, c'est précisément cette information qui a permis d'éviter l'utilisation des Thread. Pour nous un cycle correspond

à une exécution de la boucle principale et dans cette boucle nous faisons évoluer chaque agent une fois (figure 4).

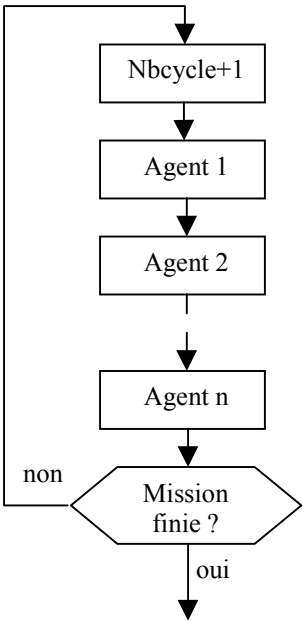

figure 4 : Algorithme n'utilisant pas de thread

La figure 5 représente un des écrans de simulation avec deux restaurants, trois aveugles et autant de paralytiques. Les petits carrés rouges représentent chacun un aveugle. Les carrés rouges et marrons représentent les restaurants et leurs zones d'appel lorsque l'aveugle est seul. La zone d'appel pour un aveugle et un paralytique, qui est plus grande n'est pas affichée ici pour ne pas surcharger le graphique. Et enfin les grandes zones dégradées du bleu vers le rouge représentent chacune un paralytique et sa zone d'attraction.

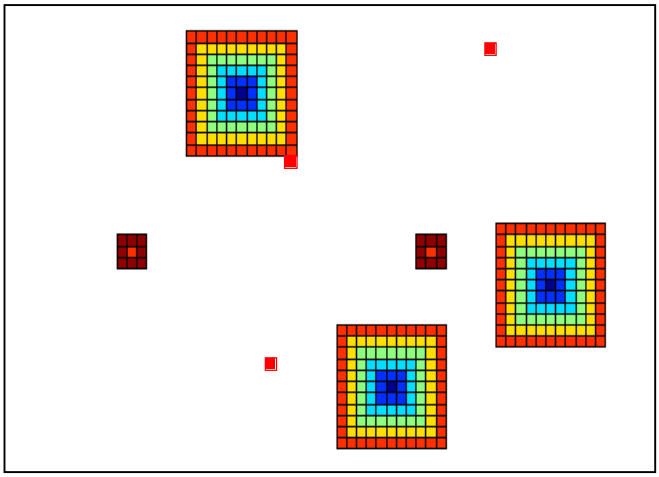

figure 5 : Exemple de simulation.

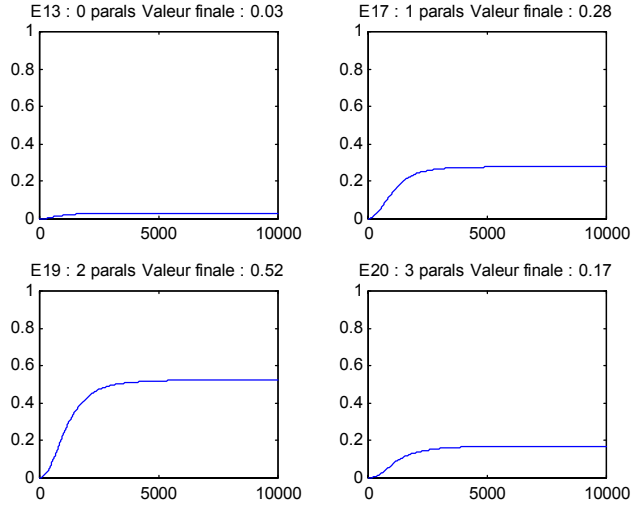

figure 6 : Résultats de simulations (3 aveugles, 3 paralytiques et 2 restaurants)

Une fois les simulations terminées, nous allons mettre en forme les résultats. Nous pourrions utiliser bon nombre de représentations. Notre but est de comparer ces résultats avec la modélisation par chaîne de Markov, nous avons donc choisi comme représentation la plus significative d'afficher la probabilité d'être dans un état en fonction du nombre de cycles. La figure 2 montre la probabilité d'avoir atteint ces états finaux (voir chapitre suivant) en fonction du nombre de cycles.

3.3 Le modèle par Chaînes de Markov

Les résultats de la figure 6 montrent un résultat de simulation. L'inconvénient de la simulation est que pour obtenir un graphique de cette précision, il est indispensable de réaliser au minimum 1000 simulations. Cela demande environ 3 heures sur un pentium 233Mhz. Nous avons donc proposé une modélisation par chaîne de Markov. L'idée principale est de limiter le nombre de simulations et de les remplacer par un modèle probabiliste, qui est nettement plus rapide.

3.3.1 La chaîne de Markov

La chaîne de Markov est un modèle probabiliste qui permet, grâce aux calculs matriciels, de connaître l'évolution d'un état dans le temps. Nous allons donc modéliser notre système en définissant un état (Ei) de la façon suivante :

Np
Na
Ng
Nr

Ei

Où

- Np est le nombre de paralytiques n'ayant pas encore été découverts.
- Na est le nombre d'aveugles dans une phase de déplacement aléatoire.
- Ng est le nombre d'aveugles, ayant trouvé un paralytique, à la recherche d'un restaurant.
- Nr est le nombre d'aveugle ayant atteint le restaurant (avec ou sans paralytique).

La figure 7 représente la chaîne de Markov pour la configuration suivante du système : 3 aveugles, 3 paralytiques, 2 restaurants. Pour plus de clarté par la suite, nous allons revenir sur quelques états importants :

- L'état E1, qui est l'état initial. Au début il y a donc 3 paralytiques, 3 aveugles, aucun paralytiques n'a été trouvé et aucun aveugles n'a atteint le restaurant.
- Les états E13, E17, E19 et E20 sont les états finaux de notre chaîne. L'état E13 correspond au cas ou les 3 aveugles trouvent seuls les restaurants. L'état E17 : 2 aveugles ont atteint seuls les restaurants et un paralytique a été découvert. L'état E19 : 2 paralytiques ont pu atteindre les restaurants et enfin l'état E20 correspond au cas ou les trois paralytique ont été découvert.

Comme il est possible de le constater sur la figure 7, avant d'atteindre les états finaux, il est indispensable de passer par une succession d'états intermédiaires qui correspondent aux différentes phases du déplacement aléatoire. Cette succession d'états ne se fait pas de façon anarchique, les flèches représentées sur la figure 7 sont les seuls moyens de passer d'un état au suivant. Et à chaque flèche est associée une probabilité, celle-ci correspond aux chances de passer de l'état précédant à l'état suivant à chaque cycle. Par exemple p1 est la probabilité de passer de l'état E1 à l'état E2. Il est important de noter que chaque franchissement de transition correspond à un temps de cycle. C'est pour cela que l'on exprime pas les probabilités globales, mais par cycle.

Une autre remarque importante, qui découle du paragraphe précédant est la suivante : étant donné que l'on travaille en temps de cycle, il n'est pas dit qu'il y ait un changement d'état à chaque fois. Cela implique que chaque état possède une probabilité de rester dans l'état courant. Ces flèches dont la sortie et l'entrée sont le même état n'ont pas été représentées sur ce graphique afin de ne pas l'alourdir.

Maintenant que nous avons décrit notre système, il ne nous reste plus qu'a déterminer la valeur de chaque probabilité, puis comparer les résultats à ceux de la simulation. Mais nous allons vite nous apercevoir que déterminer les probabilités de façon théorique n'est pas un problème aussi trivial qu'il pourrait le paraître.

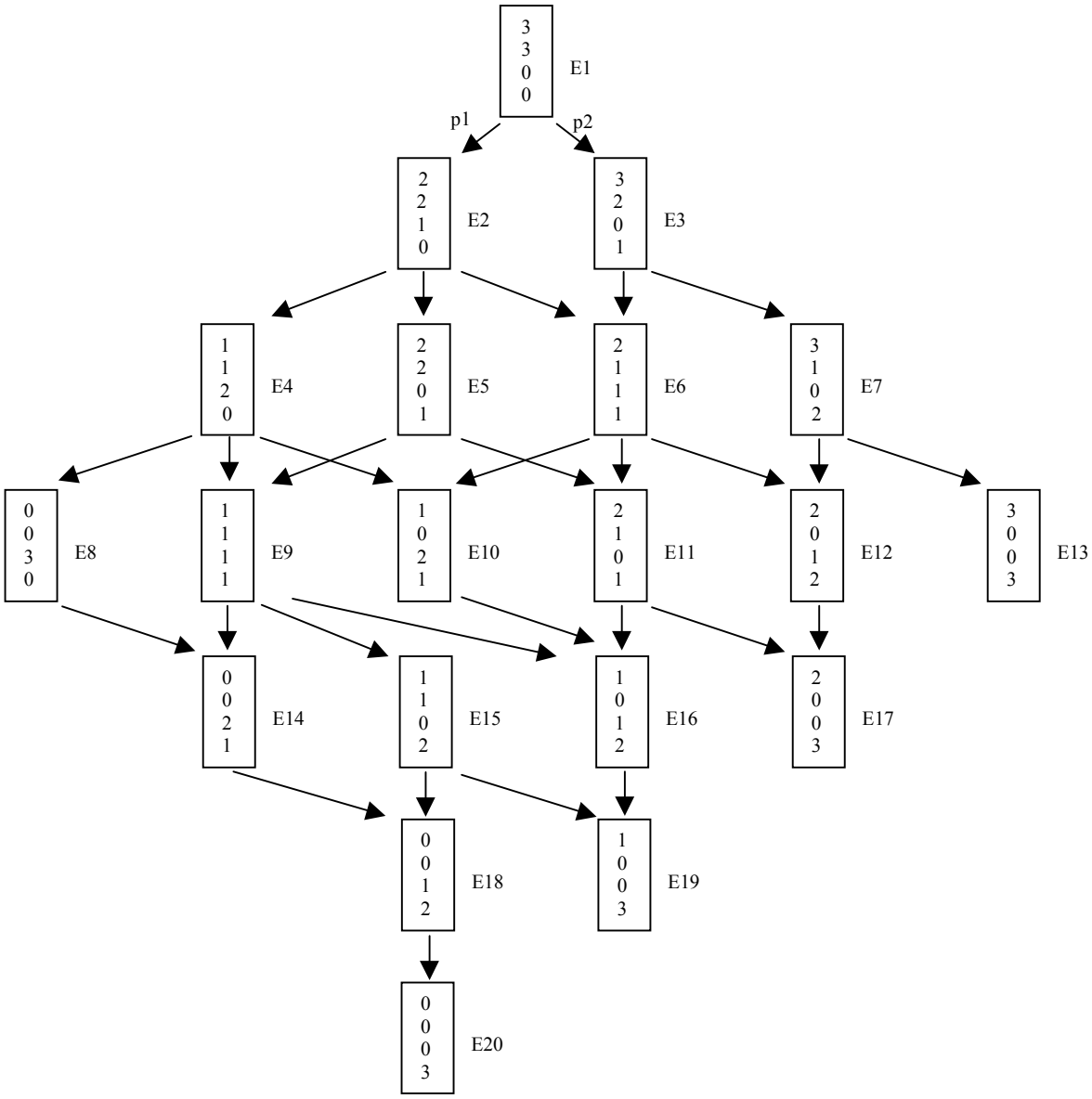

figure 7 : La chaîne de Markov pour l'aveugle et le paralytique (3 aveugles 3 paralytiques)

3.3.2. Estimation des probabilités

Pour déterminer les probabilités de changer d'état, nous nous sommes en premier lieu orienté vers une solution instinctive. La première approche a été de dire que la probabilité de trouver un paralytique (par exemple) été proportionnelle à la surface couverte par celui-ci et sa zone d'appel. Mais les premiers résultats nous ont rapidement appris deux choses : le modèle était correct, mais l'estimation des probabilités était fausse. Notre erreur venait de notre modélisation de la marche aléatoire. En effet, nous travaillons en connexité 8, et c'est cela qui fausse une grande partie de nos résultats. A. Martinolo, A. Ijspeert et F. Mondada

dans leur article [7] ne précisent pas la nature des déplacements aléatoires, mais considèrent que la probabilité d'atteindre une surface du terrain est proportionnelle à celle-ci. Cette hypothèse implique de restreindre le type de déplacements aléatoires. Or, nous souhaitions utiliser la connexité 8 qui est simple à implémenter sur un robot réactif.

Nous avons pour cela mis au point une méthode qui permet, grâce à quelques simulations de déterminer l'ensemble des probabilités. Une fois que la chaîne de Markov est étalonnée, il est possible de faire varier notre système, et d'observer les résultats de façon très rapide.

Le principe général de la méthode est de décomposer l'ensemble des probabilités en un minimum de systèmes simples. Ensuite, grâce à des combinaisons proportionnelles, on peut facilement en déduire l'ensemble des probabilités de la chaîne de Markov. Pour illustrer ce principe, prenons l'exemple de la chaîne de Markov de la figure 7. Nous avons trois aveugles, trois paralytiques et deux restaurants. Les probabilités de base seront :

- Pp : Trouver un paralytique seul.
- Pr : Trouver un restaurant seul.
- Prp : Trouver un restaurant alors que l'on a déjà trouvé un paralytique.

Ces trois cas séparés se calculeront de façon indépendante du reste du problème. En effet, nous considérerons que seuls les éléments pertinents se trouvent sur le terrain. Par exemple pour calculer Pp, nous considérons qu'il n'y a sur le terrain (qui est de même taille) qu'un aveugle et un paralytique. Notre but est de ramener chacun de ces cas particuliers à des chaînes de Markov à deux états :

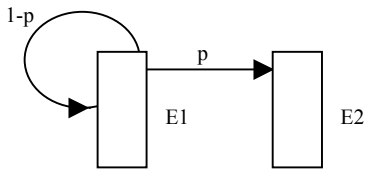

figure 8 : Chaîne de Markov à deux états

L'état E1 correspond ici à la recherche aléatoire, tant que l'aveugle n'a pas atteint sa cible, il reste dans cet état. L'état E2 est l'état final : la cible est atteinte. La probabilité de démarrer dans l'état E1 est P_{rw}. Bien évidemment, la probabilité de démarrer en E2 est $1-P_{rw}$.

Nous obtenons les systèmes suivant :

$$P = \begin{bmatrix} 1-p & p \\ 0 & 1 \end{bmatrix} \qquad S0 = \begin{bmatrix} P_{rw} \\ 1 - P_{rw} \end{bmatrix}$$

Si l'on trace l'évolution de la probabilité d'avoir atteint la cible en fonction du nombre de cycles, nous obtenons une courbe de la forme (figure9) :

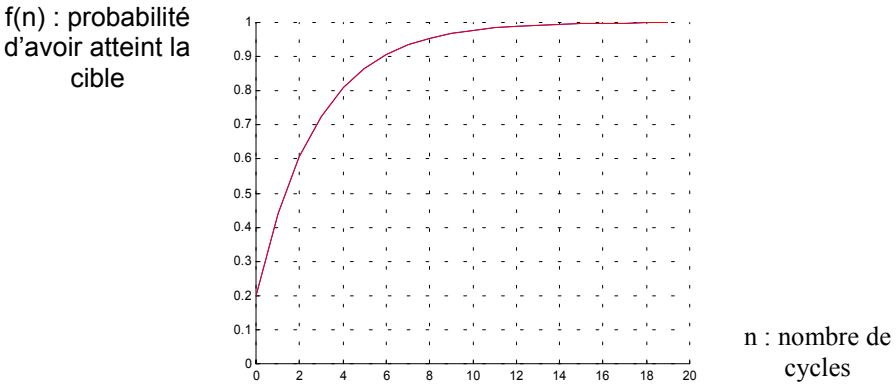

figure 9 : Evolution typique de la chaîne de Markov à deux états.(les valeurs de P et de Prw ont été choisies de façon arbitraires)

Pour le calcul de cette courbe, nous appliquons la formule de Markov d'évolution des probabilités des états suivante :

$$f_n=(P^n)^t.S_0.$$

Pour obtenir f(n), il faut diagonaliser P :

$$P=A.D.A^{-1} \text{ (avec A matrice de passage)}$$

Les valeurs propres sont :

$$\lambda_1 = \frac{-p+2+p}{2} \qquad \lambda_2 = \frac{2.(1-p)}{2}$$

Les vecteurs propres sont :

$$V_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad V_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Nous en déterminons :

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 1 & 0 \\ 0 & 1-p \end{bmatrix} \qquad A^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}$$

Ainsi :

$$f_n = (P^n)^T.S_0 = (A.D^n.A^{-1}).S_0 = \begin{bmatrix} P_{rw}(1-p)^n \\ 1-P_{rw}(1-p)^n \end{bmatrix}$$

Ainsi, nous en déduisons l'évolution temporelle de l'état E2 :

$$f(n) = 1\text{-}P_{rw}.(1\text{-}p)^n$$

Le principe général de cette méthode est de réaliser quelques simulations du cas qui nous intéresse, et d'en déterminer, à partir de la courbe, les valeurs de P et P_{rw}. Pour n=0, nous avons f(0)=1- P_{rw}. Donc f(0) nous donne la valeur de P_{rw}. Il est maintenant facile de déterminer la valeur de p à partir de n'importe quel point de la courbe. Intéressons-nous à l'exemple suivant : nous cherchons à connaître la probabilité d'atteindre un restaurant qui possède une zone d'appel de 1 avec un aveugle seul sur un terrain de 60*40. Nous avons en premier lieu réalisé des simulations afin de connaître l'allure de la courbe (Celle-ci est représentée en rouge sur la figure 5). En appliquant la méthode présentée précédemment nous en avons déduit la valeur de p : $p = 2{,}7.10^{-4}$.

Une remarque concernant P_{rw} semble s'imposer. 1-P_{rw} peut facilement se déduire de façon théorique : c'est la probabilité de commencer la recherche aléatoire en étant déjà sur la cible à atteindre. 1-P_{rw} est donc le rapport de la surface de la cible sur celui du terrain. Pour l'exemple précédent,

$$1 - P_{rw} = \frac{9}{2400}.$$

On peut constater que cette valeur va généralement être proche de zéro car la surface couverte par la cible est généralement petite comparée à celle du terrain.

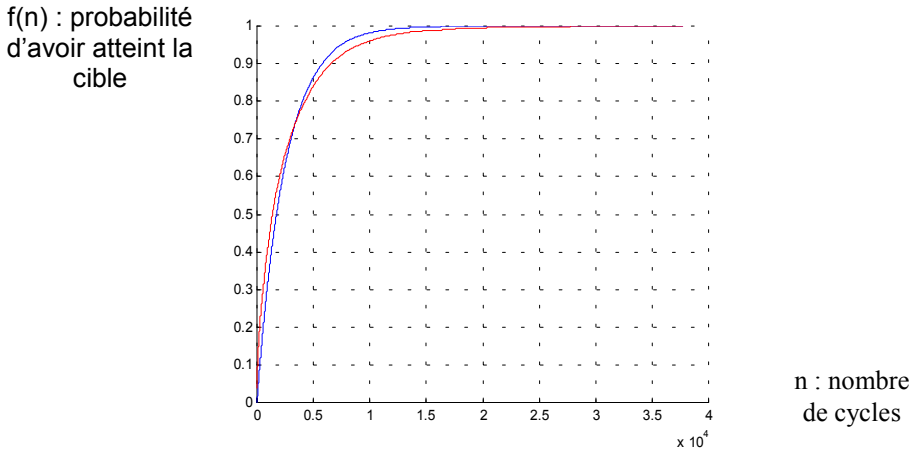

figure 10 : En rouge résultats de simulation, en bleu modèle théorique

La figure 10 représente les deux résultats, cela nous permet d'ors et déjà de comparer la théorie et la simulation. On constate tout de suite que les résultats sont corrects, l'écart est très faible et l'allure générale de la courbe théorique suit celui de la simulation. Le léger écart entre les deux courbes provient de notre modélisation, en effet, la marche aléatoire tel que nous l'utilisons ne peut pas se réduire parfaitement à une chaîne de Markov à deux états. On peut toutefois se satisfaire de ces résultats qui ne sont pas parfaits, mais suffiront amplement à déterminer l'ensemble des probabilités de notre chaîne de Markov.

3.4 Le système réel

Afin de vérifier au mieux la validité de ces travaux, nous avons mis au point un système réel. Celui-ci est composé au total de huit agents dont trois sont des robots mobiles. Ces trois agents, de conception identique représentent chacun un aveugle, ils peuvent se déplacer, entendre, s'orienter vers le bruit et différencier l'appel d'un paralytique de celui d'un restaurant. Pour des raisons techniques évidentes, les aveugles ne peuvent pas transporter les paralytiques. Alors pour pallier ce manque, sans entraver la bonne marche de l'expérience, nous avons procédé de la façon suivante : lorsque l'aveugle rencontre un paralytique, ce dernier arrête d'émettre. L'aveugle agit désormais exactement comme s'il transportait un aveugle. Cela signifie qu'il peut détecter les restaurants d'une plus grande distance. Nous allons brièvement présenter la description technique de ce système, car là n'est pas la finalité de ce rapport.

3.4.1 Les restaurants et les aveugles

Les restaurants et les aveugles se présentent sous la forme de balises. Le protocole de transmission est basé sur un signal infra-rouge modulé en fréquence. En réalité, les deux agents sont identiques, ce sont tous les deux des balises actives, à la différence prêt qu'elles émettent chacune sur une gamme de fréquence différente (1kHz pour le paralytique et 20kHz pour le restaurant). Cela étant indispensable pour permettre à l'aveugle de les différencier.

3.4.2 Les aveugles

Comme il a été précisé précédemment, les aveugles sont des robots mobiles. Un des avantages des systèmes multi-agents est de pouvoir réaliser des tâches complexes grâce à plusieurs robots réactifs. Cela permet donc d'utiliser des robots de conception simple et donc à moindre coût. C'est ce principe que nous désirions mettre en avant, et cherchions à créer des robots le moins cher possible. Nous avons donc créé une plate-forme basée sur un micro-controleur HC11 commercialisé par Motorola. Cette plate-forme représente une base indispensable au fonctionnement du robot. Elle comporte le micro-controleur, la puissance des moteurs et également la gestion des codeurs pour permettre l'asservissement. Ensuite, il est possible d'additionner des cartes en « mezzanine », et de rajouter différentes fonctionnalités au robot. Nous avons développé une carte qui nous permet la détection infrarouge.

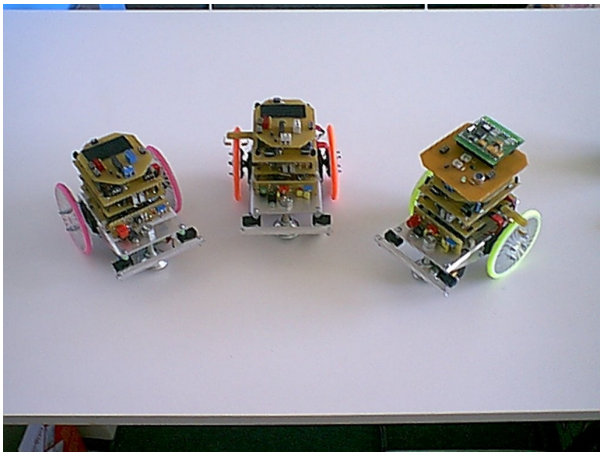

figure 11.a : Les 3 robots mobiles symbolisant les aveugles

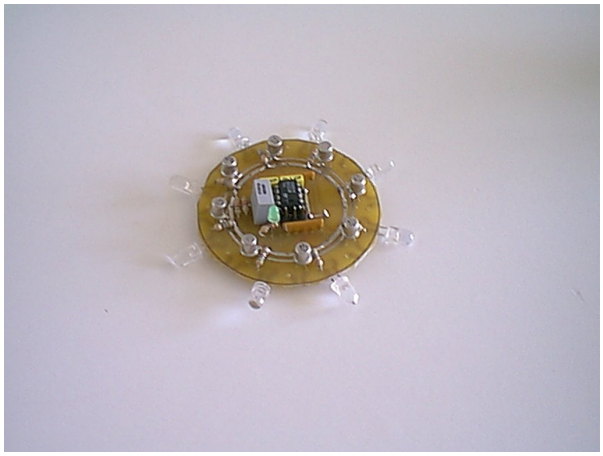

Figure 11.b : Une balise symbolisant soit un paralytique soit un aveugle selon la fréquence.

Les différentes fonctionnalités de notre agent sont les suivantes : marche aléatoire, détection de la présence d'un paralytique ou d'un restaurant dans le champ de vision et remontée du gradient. Une des difficultés de la réalisation de ce système a été la mise en place de cette dernière fonctionnalité : les capteurs infra rouge nous permettent d'avoir une information proportionnelle à la distance qui sépare celui-ci de l'émetteur. C'est grâce à une disposition judicieuse de ces capteurs que nous avons pu réaliser un asservissement qui permet de se diriger vers la cible où plus exactement de remonter le gradient.

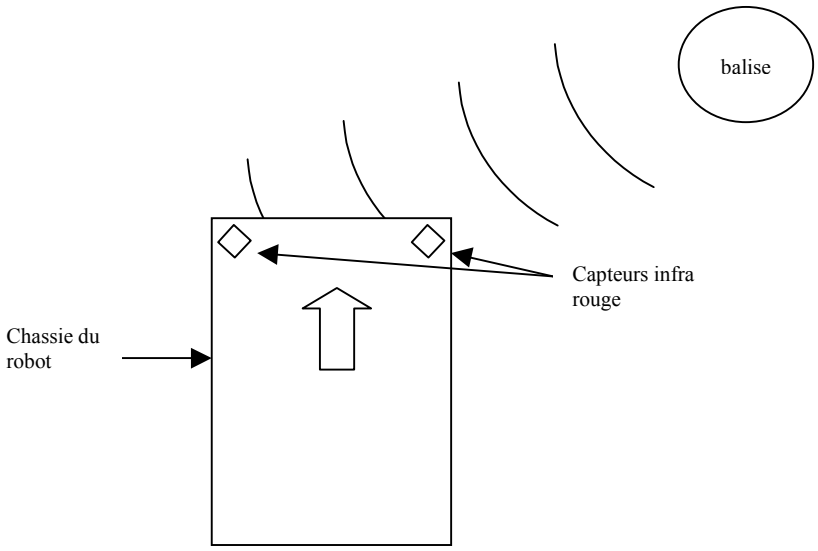

Figure 12 : fonctionnement de l'asservissement

Nous avons donc deux valeurs sur chaque capteur proportionnelles à la distance entre celui-ci et la balise. Appelons Vg et Vd les valeurs sur les capteurs respectifs : gauche et droite. L'asservissement qui sera appliqué sur les moteurs sera donc :

$$M_g = \frac{M_{\max}}{2} + (V_d - V_g).K \qquad M_d = \frac{M_{\max}}{2} + (V_g - V_d).K$$

Avec :

Mg : commande appliquée au moteur gauche.

Md : commande appliquée au moteur droit.

Mmax : consigne maximale pouvant être appliquée aux moteurs.

K : Gain proportionnel.

Ensuite un algorithme purement réactif va nous permettre de mettre en application notre système : figure 13.

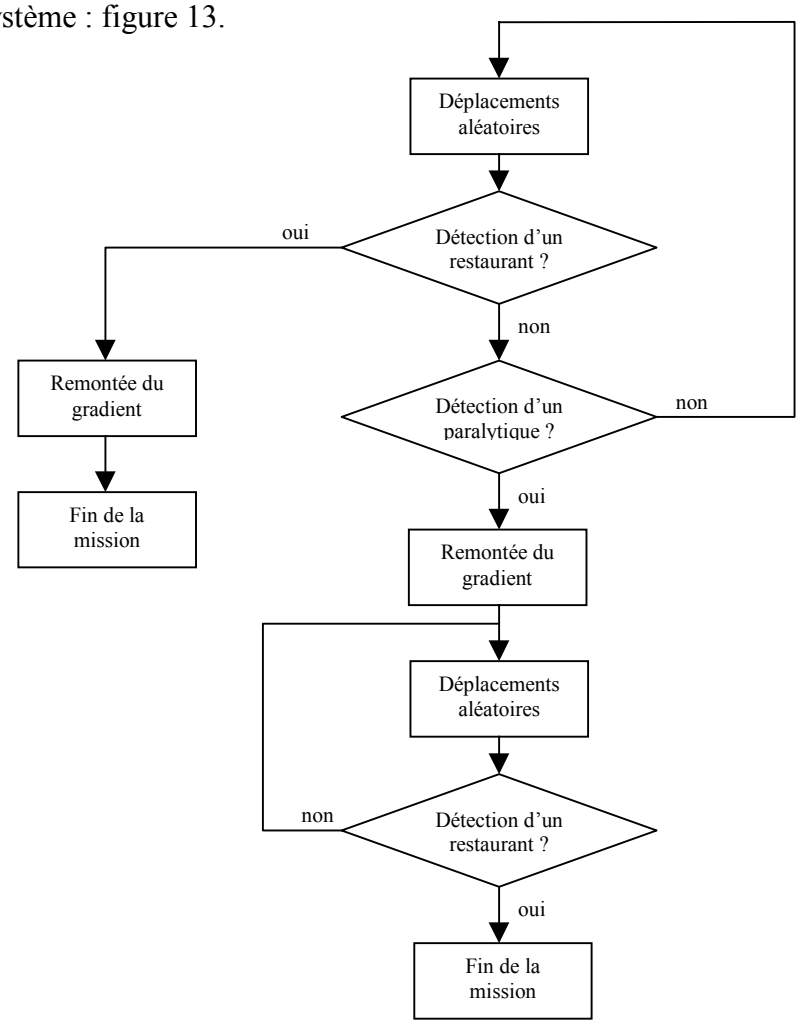

figure 13 : Algorithme des robots

Il est important de préciser que notre but n'est pas de réaliser un système qui fonctionne exactement comme le simulateur, si nous étions capable de reproduire parfaitement le fonctionnement du simulateur, cette expérience serait sans intérêt. Ce qui est justement intéressant est pouvoir observer les différences entre la théorie et la pratique. De voir comment les agents réagissent en cas de collision, suite aux erreurs accumulées par l'odométrie… Ce sont tous ces points, qui nous permettrons de conclure sur la fiabilité de notre système et de notre modèle théorique.

3.5 Conclusion

Les trois approches présentées précédemment modélisent exactement le même système : L'aveugle et le paralytique. Toutefois ces approches utilisent des méthodes

différentes, basées sur les probabilités, programmés ou réels ces trois systèmes devraient nous donner les mêmes résultats.

Toutefois, ces trois expériences possèdent chacune leurs différences, nous savons d'ores et déjà que le modèle par chaîne de Markov ne modélise pas parfaitement les résultats de simulation. Cette différence, aussi faible soit-elle, ne risque-t-elle pas d'induire une erreur qui viendra s'amplifier lorsque l'on l'introduira chacune des probabilités dans la chaîne de Markov à 20 états. On pourrait aussi se demander comment le système réagira aux variations de conditions initiales, par exemple que se passe t-il si l'on ajoute ou supprime un aveugle. L'étude qui suit à pour but de répondre à toutes ces questions.

Analyse des résultats

4.1 Introduction

Quelques expériences ont été réalisé en faisant varier le nombre d'aveugles et de paralytiques. Toutes ces expériences ont été implémentées sur le simulateur, puis modélisées grâce au modèle par chaîne de Markov. Nous discuterons des résultats entre ces expériences. Nous avons également expérimenté notre système avec trois aveugles et trois paralytiques grâce à des robots mobiles.

4.2 Comparaison Simulateur / Chaîne de Markov / Système réel

Afin de pouvoir comparer les résultats nous allons nous intéresser à l'évolution des états finaux. En effet, pour atteindre ces états dans la chaîne de Markov, il est essentiel de passer par l'ensemble des autres états. L'évolution de ces états est donc représentatif de l'ensemble du fonctionnement de la chaîne de Markov.

Nous avons réalisé sept expériences : trois simulations, trois modélisations par chaîne de Markov et une implémentation sur système réel. (tableau 1)

Terrain	Aveugles	Paralytiques	Nombre de simulations	Figure
Simulation				
60x40	3	2	10000	14a
60x40	3	3	10000	16a
60x40	4	3	10000	15a
Chaîne de Markov				
60x40	3	2	-	14a
60x40	3	3	-	16a
60x40	4	3	-	15a
Système Réel				
2.60m x 3.40m	3	3	10	16c

tableau 1 : Récapitulatif des expériences

4.2.1 Trois aveugles et deux paralytiques

Nous allons tout de suite nous intéresser au cas suivant : trois aveugles et deux paralytiques démarrent d'une position initiale aléatoire. Les résultats obtenus sont représentés sur la figure 14a. La courbe rouge en pointillée représente la simulation, et la courbe bleu la

chaîne de Markov. Ces courbes représentent la probabilité d'atteindre l'état final (E13 : 0 paral, E17 : 1 paral, E19 : 2 parals et E20 : 3 parals) en fonction du nombre de cycles. Evidement, dans le cas présent, comme il n'y a que deux paralytiques, il est impossible d'atteindre l'état E20. Celui-ci n'a pas été représenté. On constate que les valeurs des états finaux sont sensiblement les mêmes pour la simulation et la chaîne de Markov. La figure 10b représente l'erreur entre les courbes de la figure 14a. Cette erreur est très faible, toutefois, on observe des pics aux environs de 1000 cycles. Ces pics traduisent en réalité un retard entre les deux courbes. Cela est du aux hypothèses émises sur la modélisation, en effet nous modélisons un système qui devrait comporter plusieurs états par un graphe à deux états. C'est cette approximation qui engendre ce retard.

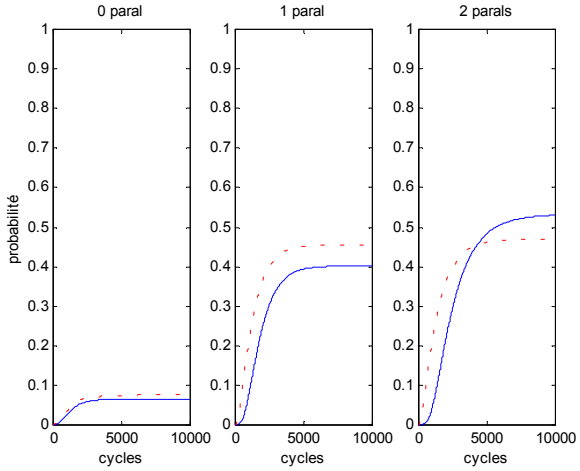

Figure 14 a: Evolution de la probabilité en fonction du nombre de cycles.

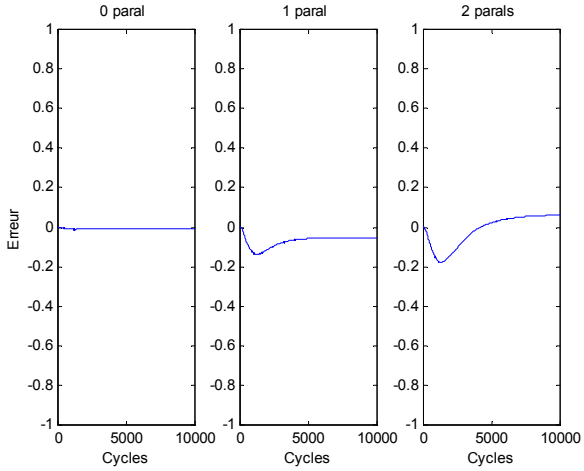

figure 14 b: Evolution de l'erreur en fonction du nombre de cycles.

4.2.2 Quatre aveugles et trois paralytiques

Nous allons maintenant nous intéresser à une autre configuration de notre système qui est la suivante : 3 paralytiques et 4 aveugles. Cet exemple est particulièrement intéressant dans le cas où nous désirons amener tous les paralytiques au restaurant. Il y a, en effet, un aveugle supplémentaire par rapport au nombre de paralytiques et cela pourrait nous amener à penser que le cas le plus fréquent correspond à l'état E20 : tous les paralytiques on été trouvés. Or, sur la figure 15a, l'on s'aperçoit que le cas le plus fréquemment rencontré est l'état E19. On remarquera que le modèle par chaîne de Markov avait correctement prédit cette évolution du système. Pour le reste, les conclusions sont identiques à celle de 4.2.1.

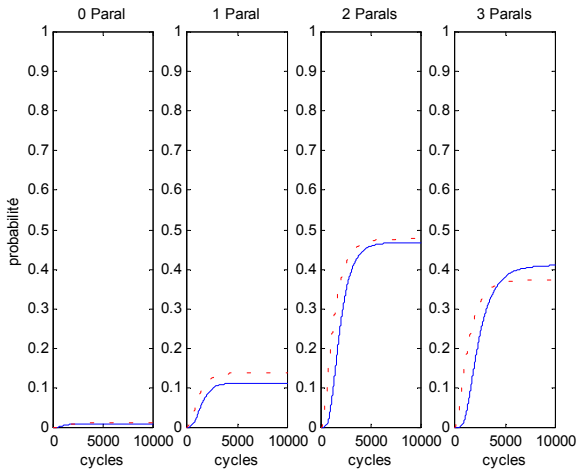

Figure 15 a: Evolution de la probabilité en fonction du nombre de cycles.

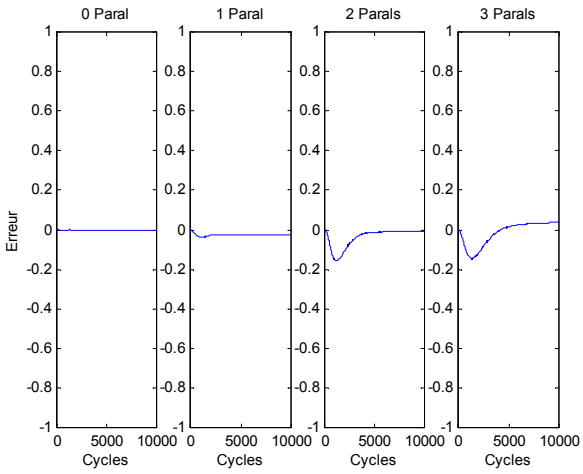

figure 15 b: Evolution de l'erreur en fonction du nombre de cycles.

4.2.3 Trois aveugles et trois paralytiques

Cet exemple est le plus intéressant, car les aveugles et les paralytiques sont en nombre égal. C'est aussi pour cela que nous avons choisi ce cas de figure afin de l'implanter sur le système réel. Contrairement au simulateur, le système réel nécessite l'intervention humaine à chaque simulation, afin d'initialiser les robots et de récupérer les paramètres stockés. C'est pour ces raisons que le nombre d'expérimentations sur le système réel est si peu élevé : 10 (figure 16c). Nous n'avons pas la prétention de valider les modèles de façon absolue en nous basant uniquement sur 10 simulations. Il est évident que le nombre de combinaisons aléatoires est très élevé et qu'il faudrait bien plus que 10 expériences pour obtenir un résultat correct. Notre but est plutôt d'obtenir la réponse à la question suivante : si nous devions réaliser le système, pouvons nous espérer obtenir des résultats se rapprochant de notre chaîne de Markov.

Lorsque le robot avance d'un pas, il parcourt 17cm, soit sa propre longueur. Si nous avions respecté les dimensions réelles du système (60x40), nous aurions dû laisser évoluer les robots sur un terrain de 6,80m x 10,20m. De plus les expériences auraient été très longues. Nous avons donc choisi volontairement de diminuer ce terrain (2,6m x 3.4m) même si, de ce fait, la simulation et le système réel durent être réalisés à des échelles différentes.

Afin de ramener tous ces résultats à la même échelle, nous avons déterminé le coefficient k qui permet de rapporter les résultats à des échelles identiques : Ce coefficient est le rapport du nombre de cases réel sur le nombre de cases utilisées :

$$k = \frac{\frac{60 \times 40}{260 \times 340}}{17^2} = 7.87$$

Les résultats de la figure 16a et 16b nous montre tout d'abords que les valeurs finales des états 13,17,19 et 20 sont bien identiques, que ces résultats soient obtenus par simulations ou par chaîne de Markov. Cela prouve que les chances d'amener les paralytiques au restaurant sont conformes à la simulation. Ensuite, les conclusions quant à l'évolution des probabilités sont identiques à celle du paragraphe 4.2.1.

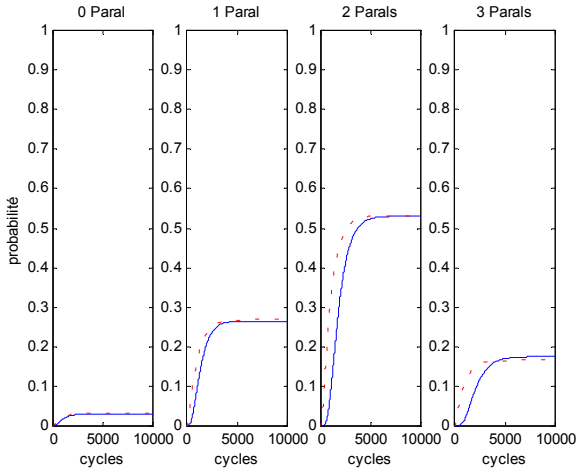

Figure 16 a: Evolution de la probabilité en fonction du nombre de cycles.

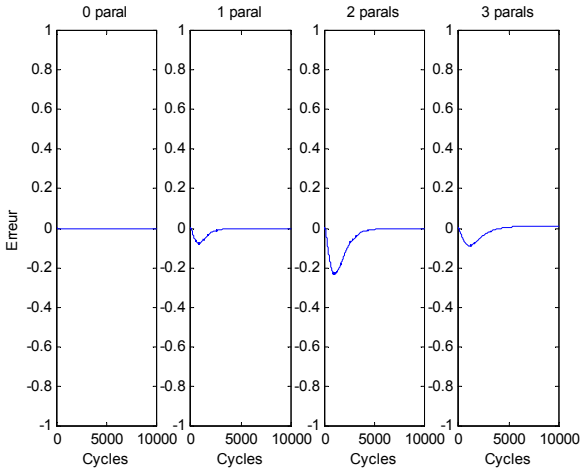

figure 16 b: Evolution de l'erreur en fonction du nombre de cycles.

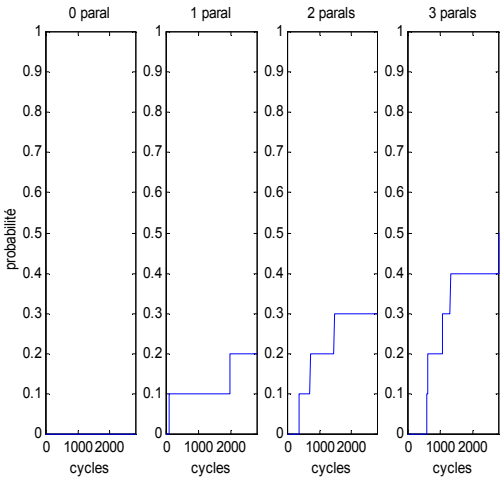

Figure 16 c: Evolution de la probabilité en fonction du nombre de cycles. (système réel)

Une remarque s'impose ici quand à la valeur des états finaux : en effet, l'état E19 est plus souvent atteint que l'état E20. Cela revient à dire que le cas le plus fréquent est de trouver deux paralytiques et un restaurant seul. Or, la surface couverte par les paralytiques est toujours supérieure à celle couverte par les restaurants au regard d'un aveugle seul. Cela nous démontre ce qui a été expliqué précédemment : les probabilités ne sont (dans notre cas) pas proportionnelles à la surface à atteindre.

17a : Début de l'expérience 17b : Fin de l'expérience

figure 17 : Photos de l'expérience

Pour conclure ce chapitre, nous allons discuter des résultats du système réel. Il est possible de distinguer sur la figure 17 deux photos de l'expérience. On distingue aisément les trois robots mobiles qui symbolisent les aveugles. Sur la photo 17b, on constate que les aveugles ont bien atteint leur cible, à savoir les deux restaurants. Pour les raisons expliquées précédemment, les paralytiques restent à leur place.

L'aspect saccadé des courbes de la figure 16c (le coefficient k a déjà été pris en compte dans les courbes) est du au faible nombre d'expériences. On constate que les valeurs finales des états sont relativement cohérentes puisque les 3 robots n'ont jamais atteints les restaurants seul et que cette probabilité est très faible : 0.03. De même pour l'état E17, puisque ce cas de figure est apparu deux fois sur dix, et que la probabilité en simulation est de 0,28. La surprise se situe plutôt au niveau des états E19 et E20, puisque les robots ont plus souvent atteint l'état E20 que l'état E19. Il est important de signaler que la différence entre les deux états n'est que de deux simulations. Puisque l'état E20 est survenu 5 fois contre 3 pour l'état E19. La différence majeure entre ces résultats survient sur l'état E20, qui est apparu 5 fois, ce qui fait de lui l'état le plus fréquent, alors que la simulation le prédisait avec une probabilité de 0.17. Comme nous l'avons précisé précédemment, le nombre de simulations ne

permet pas de conclure de façon absolue sur ces résultats. De plus nous avons placé les robots sur le terrain de façon aléatoire, mais peut-être avons nous de façon inconsciente instauré un biais dans la répartition aléatoire en plaçant les aveugles proches des paralytiques. Quant à l'évolution des probabilités en fonction du nombre de cycles, l'évolution de ces courbes laisse entrevoir, une réelle cohérence dans les résultats. En effet, les changements d'états sur la figure 16c interviennent toujours dans les plages de transitions des figures 16a et 16b.

4.3 Conclusion

Nous venons ici de décrire l'ensemble des résultats que nous avons obtenu grâce à nos différents modèles. Nous n'avons réalisé que deux variantes autour de notre modèle initial : 3 aveugles et 3 paralytiques. Ces analyses confirment la validité de notre modèle par chaîne de Markov. Nous avons vu en effet que les résultats concordaient correctement avec ceux de la simulation. Toutefois certaines erreurs très faibles persistent pendant les phases de transition, mais celle-ci disparaissent rapidement. De plus il nous faudrait bien plus d'états pour modéliser parfaitement le système, il semble difficile de réduire d'avantage la chaîne. Et ces résultats apparaissent plus que correct au regard des approximations qui ont été faîtes.

Un des points importants de ce chapitre est l'implantation sur un système réel. Il est vraiment dommage de n'avoir pas pu réaliser davantage d'expériences, d'autant plus que nous n'avons pas fait varier le nombre de paralytiques ou d'aveugles. Malgré cela, les résultats présents, laissent présager une correspondance convenable avec notre modèle par chaîne de Markov. Il semble important que les futurs travaux s'orientent d'avantage vers la validation sur système réel, mais également à visualiser le comportement du système à de grandes variations. Cela devrait permettre de déterminer les limites d'application de la modélisation par chaîne de Markov.

Conclusion

Un des points novateur de cette étude, est le mode de déplacement des robots. Dans les études précédentes, une des hypothèses de départ consistait à considérer que la probabilité d'atteindre une zone du terrain était proportionnelle à sa surface. En choisissant de faire évoluer les agents en connexité 8, nous avons rajouté une difficulté supplémentaire à notre étude. Malgré cela, la proposition que nous avons fait afin d'estimer les probabilités semble correcte, puisque les résultats généraux de la chaîne de Markov correspondent à ceux des simulations.

Malheureusement, nous n'avons pas pu réaliser d'avantage d'expérimentations et nous pensons que les futurs travaux devront déterminer les limites de validité de ce système. Par exemple en faisant varier de façon importante les paramètres de notre application. Il est également envisageable d'imaginer un autre système radicalement différent de l'aveugle et du paralytique.

Cette étude fut réalisée sur une durée de six mois, l'approche initiale devait essentiellement se baser sur une approche par simulations, éventuellement la validation sur un système différent de celui de l'aveugle et du paralytique. Les premiers résultats de simulations ont rapidement laissé présager la validité de la chaîne de Markov. L'idée d'une implémentation sur système réel a immédiatement séduit M. Liégeois. Nous avons donc entrepris l'ambitieux pari de réaliser trois robots mobiles en un laps de temps limité. Ce pari fut gagné puisque ce document contient les premiers résultats de ce système. Mais cela prouve l'efficacité et la simplicité des systèmes multi-agents. En effet les systèmes multi-agents sont généralement basés sur une approche multi-robots réactifs. Cela permet de concevoir rapidement un système basique à moindre coût, d'où peut émerger un comportement coopératif. Nous venons donc de montrer que l'approche multi-agents n'est pas une utopie, et que ces applications verront probablement le jour d'ici quelques années.

B i b l i o g r a p h i e

[1] P. Rongier et A. Liégeois, « Analysis and Prediction of the Behavior of one class of Multiple Foraging Robots with the help of Stochastic Petri Nets », *Proc 1999 IEEE Int Conf. On S.M.C.,* Tokyo,p.143-148.

[2] Jean Pierre Claris de FLORIAN, « L'aveugle et le Pralytique », Fable.

[3] A. Drogoul et J. Ferber, « From Tom Thumb to the Dockers : Some Experiments with Foraging Robots », *2nd International Conference on Simulation of Adaptative Behaviour,* Honolulu, p 451-459, décembre 1992.

[4] R.C. Arkin, « Cooperation without Communication : Multiagent Schema-Based Robot Navigation », *Journal of Robotic Systems*, Vol. 9 (3), p. 351-364, avril 1992.

[5] O. Simonin, A. Liégeois et P. Rongier, « An Architecture for Reactive Cooperation of Mobile Distributed Robots », 5th International Symposium on Distributed Autonomous Robotic Systems ,Knoxville, TN, USA ,DARS'2000.(A paraître)

[6] R.A Brooks, « Prospects for Human Level Intelligence for Humanoid Robots», P*roceedings of the First International Symposium on Humanoid Robots (HURO-96)*, Tokyo, Japon, octobre, 1996.

[7] A. Martinoli, A.J. Ijspeert et F.Mondada, « Understanding Collective Aggregation Mechanisms : From Probalistic Modelling to Experiments with Real Robots. », DARS3, 1998 , Karlsruhe , Springer Ed. pp289-298, mai 1999

[8] O. Michel. « Webots : Symbiosis between virtual and real mobile robots. » In *Proceedings of the First International Conference on Virtual Worlds*, VW'98, pages 254-263, Paris, France, July 1998. Springer Verlag.

[9] A. Martinoli, E.Franzi et O. Matthey ,« Toward a Reliable Set-Up for Bio-Inspired Collective Experiments with Real Robots*», 5th International Symposium on Experimental Robotics*, pp 597-608, Juin 1997.

[10] O. Gutknecht, J.Ferber, F. Michel **,** « MadKit: Une expérience d'architecture de plate-forme multi-agent générique », Octobre 2000, (à paraître).

Résumé

Ce document présente différentes modélisations d'un système multi-agents hétérogène : l'aveugle et le paralytique. Ce système, basé essentiellement sur la coopération entre les agents, consiste à ramener le maximum d'agents jusqu'à une cible dans un environnement inconnu. Lorsque deux agents hétérogènes coopèrent, la cible leur est plus facilement accessible. Ce système est étudié grâce à un modèle probabiliste : une chaîne de Markov. Cela permet d'obtenir des résultats très rapidement contrairement à la simulation qui demande de nombreuses heures. Nous avons simulé notre système afin de comparer les résultats probabiliste et la réalité. Et enfin, il a été implémenté sur un système réel grâce à des robots mobiles. Nous présentons ici les deux simulateurs qui ont été utilisé afin de vérifier les résultats par chaîne de Markov, avant de réaliser une brève description du système réel. Enfin, nous analysons les différents résultats obtenus par nos trois modèles.

Abstract

This document introduce many modelisations of a multi-agents system : the blind man and the paralytic . This system, essentially based on cooperation between agents, consist in getting the maximum number of agent until a target in an unknow playground. Thanks the cooperation between agent, the target can be easilly find. This system is study thanks a probabilistic model : a Markov chain. This allow us to quickly get results in comparison of the simulation witch is very slow. Whatever, we have simulate us system to compare the probabilistic results and the reality. Finally. We realized an hardware implementation thanks mobiles robots. We'll introduce here the simulators, before to realized a short description of the real system. At last, we'll analyse the results of us three models.