Calcul distribué de politiques d'exploration pour une flotte de robots mobiles

Guillaume Lozenguez, Lounis Adouane, Aurélie Beynier, Abdel-Illah Mouaddib, Philippe Martinet

Calcul distribué de politiques d'exploration pour une flotte de robots mobiles

Guillaume Lozenguez[ab]
guillaume.lozenguez@unicaen.fr

Lounis Adouane[b]
lounis.adouane@lasmea.univ-bpclermont.fr

Aurélie Beynier[c]
aurelie.beynier@lip6.fr

Abdel-Illah Mouaddib[a]
abdel-illah.mouaddib@info.unicaen.fr

Philippe Martinet[b]
philippe.martinet@lasmea.univ-bpclermont.fr

[a]GREYC, Université de Caen Basse-normandie
Campus Côte de Nacre, Bd Maréchal Juin, 14032 Caen Cedex, France

[b]LASMEA, Université Blaise Pascal
24 Avenue des Landais, 63177 Aubiere Cedex, France

[c]LIP6 - Université Pierre & Marie Curie
Boîte courrier 169, 4 place Jussieu, 75005 Paris

Soutenue par l'Agence Nationale de la Recherche à travers le projet *R-Discover*.

Résumé

*Ce papier présente une architecture multi-robots permettant une allocation automatique de plusieurs objectifs sur une flotte de robots. Le challenge consiste à rendre des robots autonomes pour réaliser coopérativement leur mission sans qu'un plan soit prédéfini. Cette architecture, appelée PRDC, est basée sur 4 modules (Perception, Représentation, Délibération et Contrôle). Nous nous intéressons plus particulièrement au module de délibération en considérant le problème des voyageurs de commerce coopératifs dans un environnement incertain. L'objectif des robots est alors de visiter un ensemble de points d'intérêt représentés dans une carte topologique stochastique (*Road-Map*). Le processus proposé pour la construction des politiques collaboratives est distribué. Chaque robot calcule ses politiques individuelles possibles de façon à négocier collectivement l'allocation des points d'intérêt entre les membres de la flotte. Enfin, l'approche est évaluée via un important nombre de simulations.*

Mots-clés : *Architecture multi-robots, Processus décisionnels probabilistes, Négociation*

Abstract

This paper presents a multi-robot architecture which permits to automatically allocate a set of exploration goals for a fleet of mobile robots. The challenge is to design autonomous robots able to cooperatively perform missions without a predefined plan. The architecture, called PRDC, is based on 4 modules (Perception, Representation, Deliberation and Control). The paper focuses on the deliberative module and addresses the cooperative stochastic salesmen problem where the goal is to visit a set of points of interest. A stochastic Road-Map *is defined as a topological representation of unstructured environment with uncertainty on the path achievement. Decision making uses a distributed computation of individual Markov Decision Process in order to allocate the set of points of interest between them. Finally, a large number of simulations permit to evaluate the proposed approach.*

Keywords: *Multi-Robot Architecture, Decision Making, Negotiation*

1 Introduction

Permettre à une flotte de robots d'être complétement autonome pour réaliser des objectifs complexes est un chalenge difficile de la robotique mobile. Les comportements des robots doivent converger efficacement vers la réalisation de tous les objectifs tout en réagissant correctement aux événements perçus. Le besoin de réaliser des comportements complexes et variés pour des robots autonomes a conduit à la proposition d'architectures hiérarchiques [2][23][17].

Ce type d'architecture permet de séparer la prise de décision du contrôle/commande d'un robot en utilisant plusieurs niveaux d'abstraction. Généralement, ces architectures incluent un niveau fonctionnel qui supervise les perceptions et asservit le robot pour la réalisation d'une tâche, ainsi qu'un niveau décisionnel qui planifie la réalisation de la mission et supervise le niveau fonctionnel.

Le niveau décisionnel traite le problème de la construction des politiques d'actions collabora-

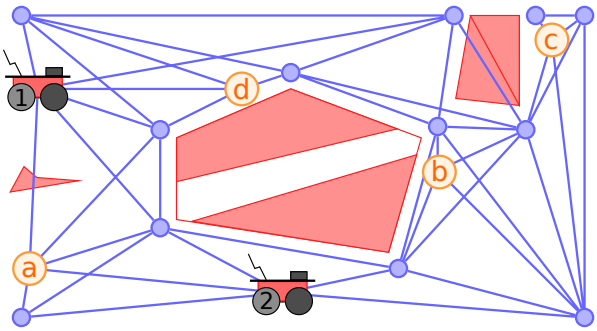

FIGURE 1 – Problématique : illustration avec 2 robots et 4 points d'intérêt $I = \{a,\ b,\ c,\ d\}$.

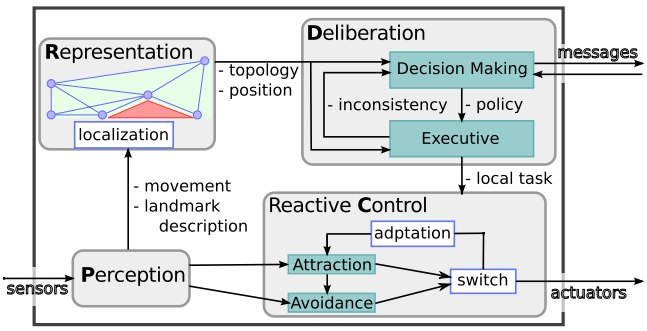

FIGURE 2 – L'architecture PRDC implémentée sur chaque robot.

tives [14][8][7] basée sur une connaissance de l'environnent modélisé par une carte. Il existe, dans la littérature, 2 principales familles de cartes qui permettent de connecter perception, décision et contrôle : les cartes sous forme de grilles d'occupation [11], qui sont une discrétisation métrique de l'environnement et les cartes dites topologiques [15] qui représentent l'environnement sous forme d'un graphe où les nœuds correspondent à des emplacements particuliers et les arcs représentent la connectivité entre les nœuds. Dans une architecture hiérarchique, nous nous intéressons à définir une carte topologique (*Road-Map*) qui intègre les contraintes de perception et qui permette la prise de décision et la supervision d'un contrôle réactif.

Dans ce papier, le niveau décisionnel est proposé pour une flotte de robots coopératifs qui ont pour mission de visiter un ensemble de points d'intérêt (Fig.1). Une solution connue consiste à utiliser les processus décisionnels de Markov décentralisés (Dec-MDPs) pour modéliser les problèmes de collaboration entre plusieurs robots [14][6]. Toutefois, calculer les politiques décentralisées optimales à partir de ce modèle est connu comme un problème difficile (NEXP-complet) [5] ce qui limite l'utilisation des Dec-MDPs au contrôle de systèmes à peu d'états.

Une coordination en ligne et à long terme de plusieurs agents oblige à utiliser des méthodes heuristiques qui conduisent à des politiques non-optimales. Une approche distribuée pour la résolution de Dec-MDPs à été proposée afin de permettre de répartir les charges de calculs sur l'ensemble des robots [8]. Même réparties ainsi, le problème du calcul des politiques décentralisées reste difficile. Par ailleurs, des approches de coordination basées sur des ventes aux enchères [10] proposent des protocoles distribués efficaces pour des problèmes d'allocation de tâches ou de ressources. Ces tâches ou ces ressources doivent alors être individuellement évaluées.

Notre approche consiste à diviser le problème du calcul des politiques décentralisées en un problème d'allocation d'objectifs couplé à un problème de calcul de politiques individuelles. Etant donnée une allocation définie à partir des intérêts individuels, chaque robot est alors capable de calculer sa politique individuelle en considérant uniquement les objectifs qui lui sont alloués. Le calcul de ces politiques permet à chaque robot d'évaluer l'intérêt de chacun des points à visiter et d'actualiser ainsi l'allocation courante.

Le papier est organisé comme suit : la section 2 décrit l'architecture des robots permettant la construction des politiques d'actions (section 3). La section 4 présente notre méthode distribuée pour l'allocation des points d'intérêt suivie par des simulations dans la section 5.

2 Une architecture délibérative

Dans cette section, nous présentons notre architecture nommée (PRDC) (Fig. 2) basée sur 4 modules que sont la Perception, la Représentation, la Délibération et le Contrôle. Ces 4 modules permettent de séparer les problématiques liées à différents domaines et s'organisent en deux niveaux hiérarchiques.

2.1 Perception et contrôle

Les modules de perception et de contrôle correspondent au niveau fonctionnel défini dans les architectures hiérarchiques [2][23]. Le module de perception supervise les capteurs de fa-

çon à convertir des flots données en informations utiles à la localisation et au contrôle du robot. Il permet d'avoir une estimation sur le déplacement du robot et sur la reconnaissance des objets alentour comme la forme des obstacles. Les données odométriques, par exemple, permettent au robot de maintenir sa localisation. Ces données sont néanmoins incertaines, elles propagent en effet, l'erreur de localisation et d'orientation au fur et à mesure du déplacement. Toutefois, le robot est capable de se localiser en reconnaissant des positions particulières [15][22], ce qui permet de corriger ponctuellement les erreurs d'odométrie.

Le module de contrôle est basé sur un ensemble de contrôleurs réactifs [19][1] (évitement d'obstacles, attraction vers une cible, suivi d'un robot ou d'une trajectoire,...). Dans le cadre de la tâche de déplacement entre deux positions, nous avons choisi d'utiliser une architecture de contrôle basée sur une structure multi-contrôleurs [1] qui permet d'alterner entre un contrôleur d'attraction vers une cible et un contrôleur d'évitement. Le passage d'un contrôleur à un autre se fait via une fonction d'adaptation permettant d'éviter les à-coups sur la commande.

Cette architecture de contrôle permet au robot de rejoindre une position dans un environnement encombré sans qu'une trajectoire soit pré-calculée. Les tâches locales du module de contrôle sont envoyées par le module délibératif (Fig. 2) et correspondent à la position immédiate à rejoindre pour orienter le robot vers un objectif plus lointain.

2.2 Représentation

Au niveau supérieur, le module de Représentation est responsable des connaissances sur l'environnement accumulées par le robot. Le module de représentation est un module actif dans le sens qu'il inclut un processus de localisation et de cartographie simultanée (SLAM : Simultaneous Localization And Mapping) [15].

La connaissance est définie par rapport aux capacités de perception. Une *Road-Map* $< W,\ P >$ est définie dans notre architecture comme une carte topologique où chaque nœud $w \in W$ (way-points) représente une position particulière identifiable par le module de perception [22] et chaque arc $p \in P$ (path) représente l'existence d'un chemin entre deux way-points $(w_p,\ w'_p)$. De cette façon, le robot est capable de maintenir sa localisation dans la *Road-Map*.

Plusieurs attributs complètent la connaissance sur les chemins de façon à permettre une supervision efficace du module de contrôle. Pour un chemin $p \in P$, un vecteur $\overrightarrow{v_p}$ donne la position relative du point cible w'_p depuis le point w_p. Un attribut c_p associe au chemin p, un coût de déplacement. En considérant l'erreur possible d'odométrie pendant un déplacement, une fonction normalisée d_p (déviation) retourne la probabilité d'atteindre un autre point alentour. Enfin, un dernier attribut u_p rapporte l'incertitude liée à la connaissance du chemin dans la *Road-Map*. Ainsi u_p est définie comme la probabilité que la *Road-Map* soit modifiée pendant le déplacement de w_p à w'_p.

$$\begin{array}{l} \forall p \in P, \quad p = (w_p,\ w'_p,\ \overrightarrow{v_p},\ c_p,\ d_p,\ u_p), \\ \quad w_p,\ w'_p \in W, \quad \overrightarrow{v_p} \in \mathbb{R}^2, \quad c_p \in \mathbb{R}, \\ \quad d_p : W \to [0,1], \quad u_p \in [0,1] \end{array}$$

Modéliser la connaissance sous forme de chemins stochastiques permet de planifier un déplacement sûr et efficace entre deux positions [3]. Sur la base de ces chemins structurés comme un graphe représentant la connectivité globale, nous nous sommes intéressés, lors de la mise en place du module de délibération, à planifier automatiquement des comportements coopératifs pour les robots de la flotte.

2.3 Délibération

Le module délibératif, second composant du niveau supérieur, connecte la représentation de l'environnement, les objectifs et le contrôle effectif du robot. Il est divisé en 2 parties : construction de la politique (Decision Making) et exécution (Fig. 2). La partie exécutive supervise le contrôle du robot par rapport à la politique d'actions calculée.

Une politique est une fonction $\pi : S \to A$ qui associe une action à chaque état défini $s \in S$. Les états du robot sont composés de l'état perceptif (qui inclut la position du robot dans sa *Road-Map*) et un état logique qui correspond à l'étape achèvement des objectifs. Etant données les capacités perceptives de nos robots, un état ne peut pas toujours inclure les positions des autres robots dans l'environnement. Les actions sont définies par rapport aux tâches réalisables par le module de contrôle. Elles sont définies par l'ensemble des chemins P.

La politique calculée dans notre approche est actualisée, en ligne, chaque fois qu'une incohérence est détectée entre la topologie locale et la politique pré-calculée. Cela se produit si la *Road-Map*, au moment de l'exécution, devient suffisamment différente de celle utilisée lors de la construction de la politique. Notons que la politique d'un robot doit être collaborative, elle peut être calculée en utilisant des protocoles décentralisés basés sur la communication.

La partie d'exécution évalue l'état courant, vérifie la cohérence entre l'action donnée par la politique et la topologie courante de façon à demander ou non une actualisation de la politique puis, transforme cette action en une tâche locale pour le module de contrôle.

3 Construction des politiques

Pour chaque robot, la partie qui consiste à construire sa politique collaborative doit permettre à la partie exécutive de superviser le contrôle du robot et d'atteindre les objectifs propres du robot. Dans ce papier nous nous intéressons au problème d'une équipe de voyageurs de commerce pour lesquels l'objectif est de visiter collectivement un ensemble de points d'intérêt. En raison de l'aspect non déterministe de la réalisation des actions de déplacement, nous avons choisi de modéliser le problème à l'aide du formalisme des Processus Décisionnels de Markov (Markov Decision Process, MDP) [18].

3.1 Les Processus Décisionnels de Markov

Un MDP est défini par un tuple $< S, A, t, r >$ avec S et A représentant respectivement les ensembles d'états et d'actions qui définissent le système et ses possibilités de contrôle. t est la fonction de transition définie par $t : S \times A \times S \rightarrow [0, 1]$ qui donne la probabilité $t(s, a, s')$ d'atteindre s' depuis s en exécutant l'action a. La fonction de récompense r est définie par $r : S \times A \rightarrow \mathbb{R}$, où $r(s, a)$ retourne la récompense obtenue en exécutant a depuis s.

Trouver la solution optimale d'un MDP consiste à chercher la politique optimale π^* qui maximise les gains espérés sur les récompenses. π^* maximise la fonction de valeur $V^\pi : S \rightarrow \mathbb{R}$ donnée par l'équation de Bellman [4]. Soit, pour une politique π :

$$V^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} t(s, \pi(s), s') V^\pi(s')$$

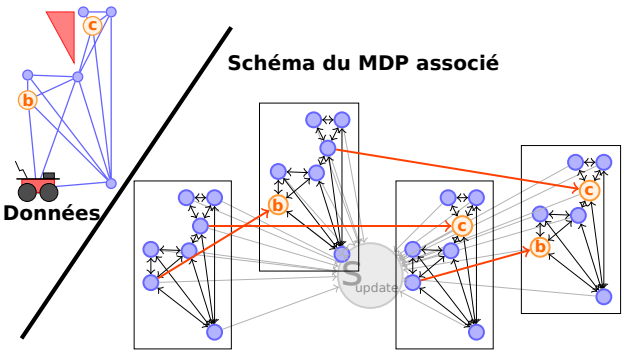

FIGURE 3 – Exemple d'une énumération des états possibles pour un MDP individuel orienté par 2 objectifs.

$$\pi^*(s) = \underset{a \in A}{argmax}(V^{\pi^*}(s))$$

Le paramètre $\gamma \in [0, 1]$ pondère l'importance entre les récompenses immédiates et futures. Dans le cas d'un problème de taille finie, comme atteindre un nombre donné d'objectifs, γ est considéré égal à 1. L'algorithme "*value iteration*" [18] permet de converger sur la politique optimale π^* et de calculer la fonction de valeur associée V^{π^*}.

3.2 MDP individuel d'exploration

Les MDPs ont été utilisés avec succès en robotique mobile pour une décision et un contrôle basés sur une représentation en grille d'occupation [12][7][21]. L'approche proposée dans ce papier se base sur une carte topologique, la politique doit assigner un point cible à atteindre au module de contrôle pour toutes les positions possibles du robot.

Un état s du MDP individuel d'exploration inclut la dernière position reconnue du robot $w_s \in W$ et l'ensemble des points d'intérêt déjà atteints I_s. Un état spécifique s_{update} est ajouté pour modéliser l'état particulier où la perception de l'environnement conduit à une incohérence de la politique. Une action a est ajoutée pour chaque chemin $p_a \in P$ pour cibler le waypoint w_{p_a}. L'action de rester sur place est modélisée comme un chemin p_s depuis w_s qui boucle sur lui même. Ce type d'action permet aussi de valider la visite d'un point d'intérêt associée à l'obtention d'une récompense ponctuelle.

$$S = \{(w_s, I_s) \mid w_s \in W, \quad I_s \subset I\} \cup \{s_{update}\}$$
$$A = \{\ (p_a) \mid p_a \in P\ \}$$
$$\cup \quad \{\ (p_s) = (w_s,\ w_s) \mid w_s \in W\ \}$$

Quand une action $a = p_a$ est exécutée depuis un way-point w_s, la fonction de transition t retourne la probabilité u_{p_a} de tomber dans l'état s_{update} ou la probabilité d'atteindre un way-point $w \in W$ en accord avec les valeurs de la fonction d_{p_a}. Une transition déterministe est ajoutée pour l'action de valider la visite d'un point d'intérêt $a = p_s$, elle conduit dans l'état correspondant où le point d'intérêt courant est ajouté à l'ensemble des points d'intérêt déjà visités.

$$t((w_s, I_s), p_a, s_{update}) = u_{p_a} \quad \mid \ (w_s = w_{p_a})$$
$$t((w_s, I_s), p_a, (w_{s'}, I_s)) = (1 - u_{p_a}) * d_{p_a}(w_{s'})$$
$$t((w_s, I_s), p_s, (w_s, I_s \cup (I \cap w_s)) = 1$$

La fonction de récompense retourne la valeur du coût c_p de suivre le chemin p et une valeur constante du gain g si un nouveau point d'intérêt est visité.

$$r((w_s,\ I_s),\ p_a) = c_{p_a}$$
$$r((w_s,\ I_s),\ p_s) = g\ if\ w_s \in I - I_s,\ else\ 0$$

Le nombre d'états augmente exponentiellement par rapport au nombre total de points d'intérêt à visiter ($|S| > 2^{|I|}$).

3.3 Politiques décentralisées

Un MDP permet de calculer une politique pour un agent. Pour adapter ce formalisme à des systèmes multi-agents coopératifs, Bernstein et al. [5] ont défini les MDPs Décentralisés (Dec-MDPs). Un Dec-MDP est défini par un tuple $< S, A, t, \Omega, O, r >$. Les éléments S, A, t et r sont similaires aux éléments d'un MDP classique avec comme différence que chaque action $a \in A$ est l'union des actions réalisées simultanément par tous les agents. L'ensemble A est un produit cartésien $A_1 \times \cdots \times A_n$ des actions individuelles. Le formalisme des Dec-MDP inclut la notion d'observation locale de l'agent où Ω est l'ensemble des observations jointes $o \in \Omega$ et la fonction O connecte les observations des n agents à la dynamique du système. $O(s, a, s', o =< o_1 \ldots o_n >) \in [0, 1]$ donne la probabilité que les agents $Ag_1 \ldots Ag_n$ observent respectivement $o_1 \ldots o_n$ quand l'action jointe a est exécutée à partir de s et que s' est atteint.

Résoudre un Dec-MDP est connu pour être difficile (NEXP-complet). Afin de gérer cette complexité, ce formalisme a été étendu pour plusieurs sous-classes de problèmes [14]. Des approches approximées ont également été proposées [14]. Parmi elles [6, 8] permettent aux agents d'actualiser itérativement leur politique pour converger sur une collaboration localement optimale.

Le nombre d'états du problème de la visite d'un ensemble de points d'intérêt par une flotte de plusieurs robots ne permet pas une résolution qui garantisse l'optimalité. Par rapport au sujet traité ici nous avons choisi d'utiliser une collection de MDPs individuels plutôt que de chercher à résoudre un Dec-MDP global. Il est considéré que seulement l'allocation de l'ensemble des points d'intérêt modifie la politique individuelle d'un robot. De cette façon, pour une allocation donnée, la connaissance de l'état des autres robots à chaque instant n'est pas requise et le MDP individuel est totalement observable. Le calcul des politiques décentralisées peut être décomposé en deux problèmes dépendants l'un de l'autre qui sont l'allocation des points d'intérêt et le calcul d'une politique individuelle. Les hypothèses utilisées ne permettent pas une résolution optimale du problème mais la résolution du probléme doit permettre aux robots de collaborer en cours de mission.

3.4 Coopération par négociation

Chaque robot, en utilisant un MDP individuel d'exploration, est donc capable de calculer sa politique d'exploration pour une attribution de points d'intérêt fixée. Au regard de cette politique et de l'équation de Bellman, le robot est capable d'évaluer son attribution et donc de la comparer avec d'autres attributions candidates. Nous avons donc mis en place un protocole de négociation basé sur la communication qui permette aux robots d'échanger leurs préférences de façon à construire une allocation des points d'intérêt qui fasse consensus.

Il existe des protocoles distribués de coopération basés sur des ventes aux enchères. Dans "Contract net" [9], un objet, une ressource, ou une tâche à réaliser est mise en vente par un agent qui devient le "manager" pour l'item ciblé, les autres agents endossent le rôle de "contractuel potentiel" et peuvent émettre des enchères. Après un temps déterminé par le "manager" l'item est attribué à l'agent ayant fait la meilleure offre. Chaque agent peut être

"manager" ou "contractuel potentiel" et plusieurs ventes peuvent s'effectuer en parallèle. D'autres protocoles sont dérivés de "Contract net" comme MURDOCH [13] qui présente des garanties de robustesse.

La coopération basée sur des ventes aux enchères est utilisée avec succès en robotique [10] ainsi qu'en robotique mobile avec des valeurs évaluées par MDP [7][20]. Cette approche ne garantit pas que les politiques décentralisées construites soient optimales. Par contre, ce type d'approche permet de distribuer la charge de calcul et de relâcher la contrainte d'une connaissance identique pour tous les agents. En effet, chaque politique est construite par l'agent depuis sa connaissance courante et individuelle de l'environnement.

Dans le cadre de l'allocation de tous les points d'intérêt pour une flotte de robots explorateurs la valeur d'un point d'intérêt pour un robot dépend des autres points qui lui sont attribués ou non. Par exemple, la valeur du point d'intérêt b pour le robot 2 n'est pas la même s'il doit ou non visiter c (Fig.1). L'utilisation des protocoles classiques de ventes induit une remise aux enchères constante de tous les points d'intérêt jusqu'à obtention du consensus. Aussi, par la suite nous proposons un protocole basé sur le principe d'une table ronde où, à chaque itération, tous les points d'intérêt sont réévalués en parallèle et seuls les points d'intérêt pour lesquels une meilleure allocation est trouvée sont échangés.

4 Allocation des points d'intérêt

Nous nous intéressons particulièrement aux étapes de la mission où plusieurs robots de la flotte communiquent pour calculer leur politique collaborative. La communication est considérée sans erreur pendant ces étapes. Dans un premiers temps, nous nous intéressons à l'évaluation individuelle des points d'intérêt puis au protocole de table ronde qui permet aux robots de trouver un allocation qui fasse consensus. Cette approche s'oppose à une résolution centralisée sur un leader [16].

4.1 Evaluation individuelle

Un robot Ag_i est capable de construire son MDP individuel d'exploration à partir de sa propre *Road-Map* en considérant qu'il est seul pour visiter tous les points d'intérêt (I). A partir de l'ensemble des points d'intérêt qui lui sont attribués (I_{Ag_i}) et de sa position courante dans sa carte (w_{Ag_i}), le gain individuel (gi) du robot Ag_i est défini en considérant que tous les points d'intérêt qui ne lui sont pas attribués sont visités ($(I - I_{Ag_i})$). Le gain individuel est la valeur donnée par l'équation de Bellman pour l'état $(w_{Ag_i},\ I - I_{Ag_i})$.

$$gi(I_{Ag_i}) = V^{\pi^*}(w_{Ag_i},\ I - I_{Ag_i})$$

Ainsi formalisé, le gain individuel exprime la somme cumulée des récompenses qui seront statistiquement perçues par le robot Ag_i pour la visite des points d'intérêt appartenant à I_{Ag_i}. En considérant le robot comme faisant partie d'une équipe, il est intéressant de répercuter une valeur sociale représentant un coût occasionné par les actions d'un robot sur son groupe. La difficulté consiste à distribuer sur chacun des robots présents des contraintes globales. Par exemple, équilibrer le nombre de points d'intérêt attribués sur les n robots présents.

Un gain espéré (ge) propre à chaque robot et à sa charge d'exploration est donc défini comme la différence du gain individuel et du coût social occasionné. A partir de ce gain espéré, il est possible d'évaluer l'utilité d'ajouter ou d'enlever un point d'intérêt $k \in I$ à I_{Ag_i}.

$$ge(I_{Ag_i}) = gi(I_{Ag_i}) - |\frac{|I|}{n} - |I_{Ag_i}|| * oc$$

$$utility(k, I_{Ag_i}) = eg(I_{Ag_i} + k) - eg(I_{Ag_i} - k)$$

Le coût social proposé ici, est proportionnel à la différence entre : la taille de l'allocation propre au robot Ag_i et la taille moyenne ($|I|/n$ pour n robots). Le facteur oc définit le coût à opposer au gain individuel. Ainsi, chaque robot s'autorise à déséquilibrer plus ou moins sa propre allocation en comparant ses utilités qui sont fonction de sa charge actuelle, avec les utilités communiquées par ses collaborateurs.

4.2 Protocole de la table ronde

Les n robots présents lors d'une étape de communication souhaitent donc se répartir entre eux l'ensemble des points d'intérêt à visiter. L'approche distribuée ici consiste à permettre à chaque robot de construire en parallèle sa propre allocation. La difficulté réside dans le fait que la valeur d'utilité d'un point d'intérêt pour un agent est réévaluée à chaque fois que l'agent effectue une modification sur son allocation. Ce

constat nous conduit à proposer un protocole dit de la table ronde *roundTable* où toutes les allocations s'effectuent et se terminent en même temps.

Le protocole *roundTable* se divise en 5 étapes :

1 - Inscription La table est ouverte à l'initiative d'un agent, la première phase vise à recenser les agents qui feront partie de la négociation. Il n'est pas possible pour un agent d'entrer ou de sortir simplement de la table à partir du moment où l'inscription est close. Cette étape permet notamment d'identifier le réseau de communication.

Dans le cadre de notre étude, nous considérons un groupe de robots où chaque robot peut communiquer directement avec chacun des autres robots.

2 - Définition des règles Les agents inscrits doivent ensuite valider les règles de fonctionnement de la table ainsi que les règles d'attribution finale des items. Les règles de fonctionnement peuvent intégrer une attribution de rôles particuliers utiles au bon fonctionnement de la table (comme un président).

Dans le cadre de notre étude, les règles de fonctionnement sont connues initialement par les agents et elles ne nécessitent pas de rôle particulier. La règle d'attribution affecte chacun des points d'intérêt au robot qui en propose la plus forte valeur d'utilité.

3 - Enumération des items Les items sont, par la suite, énumérés de façon à ce que chaque agent puisse construire son tableau de valeur (Item/Agent).

Ici, chaque robot énumère l'ensemble des points d'intérêt dont il a la charge. Nous supposons que tous les points d'intérêt sont identifiables dans les cartes individuelles.

4 - Itérations Tous les agents actualisent et communiquent leurs valeurs conformément aux règles de fonctionnement établies.

Ici, les robots sont désynchronisés, ils itèrent et communiquent en parallèle sur un modèle de broad-cast. Chaque robot communique son vecteur de nouvelles valeurs à tous les autres robots.

5- Fermeture Les itérations sur les valeurs s'effectuent jusqu'à trouver un consensus, c'est-à-dire que pour chaque agent l'allocation donnée par les règles d'attribution ne modifie plus ses valeurs communiquées. L'allocation peut donc être validée et la table fermée. La garantie d'atteindre cet état n'est pas triviale, elle dépend des règles de fonctionnement établies et de la cohérence des comportements individuels.

4.3 Comportement individuels

Le protocole *roundTable* pour l'attribution des points d'intérêt est indépendant des comportements individuels permettant aux agents d'évaluer leurs utilités. Par contre, fermer la table sur un consensus général suppose certaines capacités de la part des agents présents : la capacité individuelle d'un agent à proposer un vecteur de valeurs d'utilité stable quelles que soient les valeurs d'utilités communiquées par les autres ; la capacité sociale des agents à proposer des valeurs d'utilité qui permettent de converger sur un consensus.

La fonction d'utilité proposée (Section 4.1) est cohérente sur les valeurs calculées. Pour un robot, quelque soit un point d'intérêt, ajouter ou supprimer ce point dans son attribution courante ne modifie pas la valeur de ce point. Par exemple (Fig.1) la valeur d'utilité du point a pour un robot est la même en considérant les attributions individuelles $\{a, b, c\}$ ou $\{b, c\}$. Cette cohérence permet aux robots d'avoir une configuration lui permettant d'être individuellement stable quelque soit les utilités communiquées par les autres robots.

Le comportement de chaque robot vise à maximiser une somme globale de gain espéré sur tous les robots. Bien que cela permette *a priori* de converger sur un consensus, chaque robot est limité à un nombre fini d'actualisation sur ses valeurs de façon à garantir l'arrêt de la table ronde. De cette façon, le consensus est forcé après avoir dépassé un certain temps. De plus, l'état stable ou instable de chaque robot est explicitement communiqué pour détecter l'état global de consensus.

En considérant des communications sans erreur et sans perte, les robots sont capables individuellement de calculer l'allocation et de détecter la fermeture de la table. Les robots utilisent actuellement un même protocole qui consiste à :

– actualiser le tableau des valeurs d'utilités grâce aux communications,
– s'il existe un ou des points d'intérêt appartenant à seulement un des ensembles constitués : 1- par l'attribution courante du robot, 2- à partir des règles d'attribution (mais pas aux deux) alors, détecter tous ces points et com-

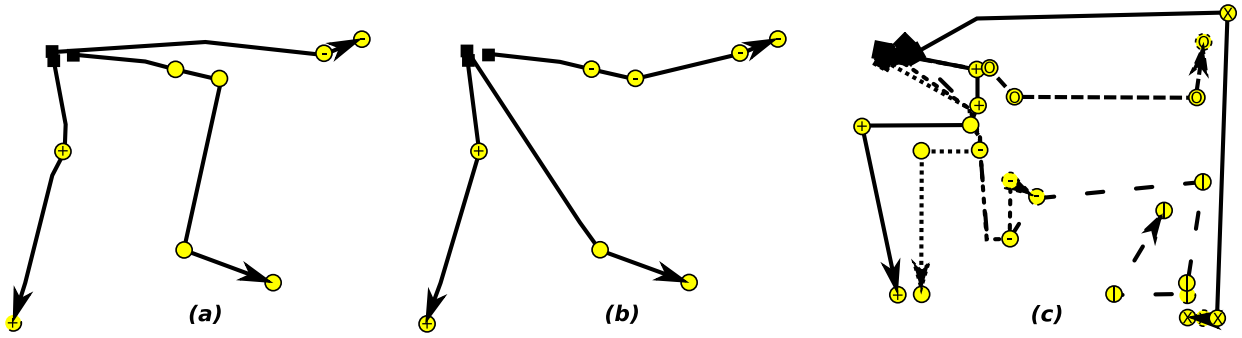

FIGURE 4 – les 2 *Road-Map* considérés : (a)(b) avec quelques obstacles et (c) le labyrinthe. (a)(c) présentent une allocation construite de façon distribuée et (b) l'allocation optimale.

muniquer un état instable ; sinon communiquer un état stable et sauter l'étape suivante,
– ajouter ou enlever dans l'attribution courante, le point le plus intéressant parmi les points détectés et communiquer les nouvelles valeurs,
– boucler jusqu'à détecter l'état de consensus.

Ce comportement individuel (dit bavard) est défini pour un cadre où les communications sont peu coûteuses. Il est possible, pour minimiser les communications, d'instaurer un tour de parole et/ou de limiter les valeurs communiquées aux valeurs qui correspondent à un état stable du robot.

TABLE 1 – Scores

$\|I\|$	quelques obstacles, $oc = 0$				quelques obstacles, $oc = 5.12$			
	Ran.	Dis.	Wo.	Po.	Ran.	Dis.	Wo.	Po.
2-3	42.1	99.5	85.8	89.5	49.7	99.6	80.0	92.5
4-5	40.6	99.0	77.5	80.3	47.4	99.0	47.4	69.3
6-7	37.5	98.9	83.3	71.8	54.6	99.2	82.6	47,8
8-9	36.8	98.4	86.1	63.8	64.7	99.2	86.5	45.0
10-11	34.1	98.1	72.7	57.0	71.3	99.3	87.6	44.5
12-13	32.0	98.4	83.9	55.7	76.0	99.2	91.6	37.0
	labyrinthe, $oc = 0$				labyrinthe, $oc = 5.12$			
2-3	40.8	99.7	89.0	97.8	48.3	99.5	78.7	86.8
4-5	44.6	99.7	77.8	85.5	50.9	99.7	88.8	63.0
6-7	39.1	99.5	81.4	73.6	54.8	99.2	75.8	42.0
8-9	37.9	99.3	85.5	67.2	54.1	99.0	75.0	34.0
10-11	36.0	99.0	85.1	53.8	58.8	98.8	82.8	32.3
12-13	36.2	98.8	83.2	44.5	64.9	98.8	86.8	23.0

(Ran.) et (Dis.) : scores moyens (%) des allocations aléatoires et construites par négociation, (Wo.) le pire score obtenu (%) et (Po.) le pourcentage de simulations où l'allocation obtenue est l'optimale.

5 Résultats de simulation

Une première série de simulations cherche à évaluer l'efficacité du protocole d'allocation distribuée par rapport à une solution optimale calculée de façon centralisée par une recherche exhaustive. L'optimalité est définie comme l'allocation qui maximise la somme des gains espérés de tous les robots.

Les simulations considèrent 3 robots dans 2 cartes différentes (Fig. 4) et entre 2 et 13 points d'intérêt. Pour 13 points d'intérêt, cela induit 3^{13} allocations possibles et entre $50 * 2^{13}$ et $70 * 2^{13}$ états pour les MDPs individuels (en fonction de $|W|$). Les coûts sur les chemins sont calculés directement comme la distance entre 2 positions de la carte ; un coût maximum de 51.2 est relevé pour la diagonale des limites des environnements. La récompense gagnée pour visiter un point d'intérêt est posée à 1000 et le coût d'opportunité oc est égal à 0 ou à 5.12 ($0.1 * maxCost$).

En se basant sur la somme des gains espérés, un score (en pourcentage) peut être calculé pour chaque allocation proportionnellement aux valeurs de la pire et de la meilleure allocation possible. Par exemple, une des simulations (labyrinthe, $|I| = 4$, $oc = 0$) a donnée les valeurs de 3891.11 (pour la meilleure) ; 3836.41 (pour la pire) ; 3845.74 (pour une allocation aléatoire) ; 3890.84 (pour l'allocation par négociation) et donc un score de 99.5%. La première série de simulations est basée sur 200 générations aléatoires sur la position des points d'intérêt pour chaque classe de simulations (type de carte, nombres de points d'intérêt et valeur de l'oc). Le Tableau 1 présente la moyenne des scores obtenus ainsi que le pourcentage des cas où l'allocation construite par négociation est la meilleure possible.

Les scores obtenus (Tableau 1) nous permettent de valider que l'allocation par une approche distribuée reste proche de l'optimale pour des problèmes de tailles réduites avec un score moyen de 99% et seulement 2 expériences qui retournent un score inférieur à 70%. D'un autre côté, la probabilité de trouver réellement l'allocation optimale diminue avec la possibilité de tomber dans des optimums locaux (surtout dans

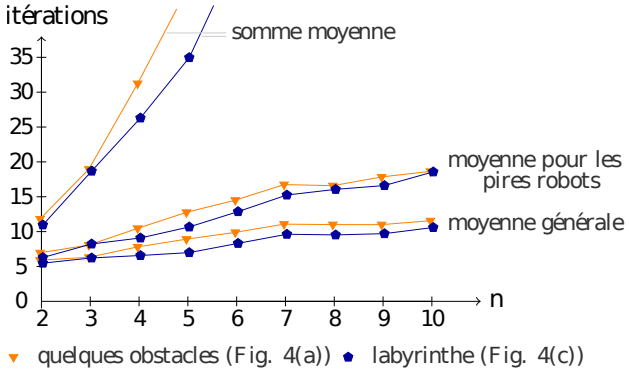

FIGURE 5 – Nombre moyen d'itérations utiles (n robots et $4 * n$ points d'intérêt).

le cas du labyrinthe). Des allocations sont trouvées avec de bons scores mais non optimaux.

Une seconde série de simulations est proposée pour valider la capacité de l'approche à traiter une flotte de robots plus importante. En considérant un coût d'opportunité de 5.12 pour équilibrer l'allocation, chaque robot peut limiter la taille de son MDP individuel en ne considérant que son attribution courante. Les simulations intègrent jusqu'à $n = 10$ robots et une moyenne de 4 points d'intérêt pour chaque robot ($|I| = 4*n$). L'allocation optimale est parmi les 10^{40} solutions possibles.

En utilisant des agents bavards, nous nous sommes intéressés au nombre d'itérations utiles par robot Ag_i où l'attribution I_{Ag_i} est actualisée (Fig. 5). Une itération utile correspond à une actualisation des utilités suivie par une phase de communication. 2 moyennes du nombre d'itérations utiles sont calculées : une à partir de tous les robots et une seconde en ne considérant que le pire des robots de chaque simulation. La figure 4(c) présente les résultats d'une simulation pour 6 robots et 24 points d'intérêt dans le labyrinthe. Il est possible de conclure que seulement quelques actualisations individuelles sont nécessaires par rapport au nombre total de points d'intérêt.

6 Conclusion

Une architecture hiérarchique basée sur 4 modules (Perception, Représentation, Délibération et Contrôle) est présentée dans ce papier. Cette architecture sépare les problèmes de l'élaboration de la décision et du contrôle/commande du robot. Nous nous sommes intéressés au module délibératif gérant la construction des politiques collaboratives permettant de visiter un ensemble de points d'intérêt avec une flotte de robots.

L'approche distribuée proposée permet aux robots d'aboutir à une solution valide. Un important nombre de simulations permet une évaluation encourageante de l'efficacité de l'approche sur des problèmes de tailles restreintes et démontre, d'autre part, la capacité de l'approche à considérer un nombre relativement important de robots. Ces résultats sont encourageant sachant que le problème traité ne peut être optimalement résolu.

Dans de futurs travaux, l'approche devrait être utilisée sur des robots réels avec une carte initiale incomplète. Nous souhaitons aussi étudier l'intérêt (efficacité, robustesse) d'utiliser différentes règles de fonctionnement pour la table ronde et différents comportements individuels.

Références

[1] L. Adouane. Hybrid and safe control architecture for mobile robot navigation. In *9th Conference on Autonomous Robot Systems and Competitions*, Portugal, May 2009.

[2] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An architecture for autonomy. *Journal of Robotics Research*, 17(4) :315–337, 1998.

[3] R. Alterovitz, T. Siméon, and K. Goldberg. The stochastic motion roadmap : A sampling framework for planning with markov motion uncertainty. In *Robotics : Science and Systems*, 2007.

[4] R. Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6 :679–684, 1957.

[5] D. S. Bernstein, S. Zilberstein, and N. Immerman. The complexity of decentralized control of markov decision processes. In *16th Conference on Uncertainty in Artificial Intelligence*, 2000.

[6] A. Beynier and A.I. Mouaddib. An iterative algorithm for solving constrained decentralized markov decision processes. In *The Twenty-First National Conference on Artificial Intelligence (AAAI-06)*, pages 1089–1094, 2006.

[7] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21 :376–386, 2005.

[8] I. Chades, B. Scherrer, and F. Charpillet. A heuristic approach for solving decentralized-pomdp : Assessment on the pursuit problem. In *SAC '02 : Proceedings of the 2002 ACM symposium on Applied computing*, pages 57–62, New York, NY, USA, 2002. ACM.

[9] R. Davis and R. G. Smith. Negotiation as a metaphor for distributed problem solving. *Artificial Intelligence*, 20 :63–109, 1983.

[10] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination : A survey and analysis. *Proceedings of the IEEE*, 94 :1257–1270, 2006.

[11] A. Elfe. Sonar-based real-world mapping and navigation. *Robotics and Automation*, 3 :249–265, 1987.

[12] A. F. Foka and P. E. Trahanias. Real-time hierarchical pomdps for autonomous robot navigation. *Robotics and Autonomous Systems*, 55(7) :561–571, 2007.

[13] B. P. Gerkey and M. J. Mataric. Sold ! : auction methods for multirobot coordination. *Transactions on Robotics and Automation*, 18 :758–768, 2002.

[14] C. V. Goldman and S. Zilberstein. Decentralized control of cooperative systems : Categorization and complexity analysis. *Journal of Artificial Intelligence Research*, 22, 2004.

[15] B. Kuipers and Y. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8 :47–63, 1991.

[16] G. Lozenguez, L. Adouane, A. Beynier, P. Martinet, and A. Mouaddib. Map partitioning to approximate an exploration strategy in mobile robotics. In *Advances on Practical Applications of Agents and Multiagent Systems*. 2011.

[17] G. Mourioux, C. Novales, and G. Poisson. Control robot by a generic control architecture. In *IEEE International Conference on Intelligent Robots and Systems*, pages 3050 – 3055, 2007.

[18] M. L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[19] J. Rosenblatt. Damn : A distributed architecture for mobile navigation - thesis summary. In *Journal of Experimental and Theoretical Artificial Intelligence*, pages 339–360, 1995.

[20] M. T. J. Spaan, N. Goncalves, and J. Sequeira. Multirobot coordination by auctioning pomdps. In *International Conference on Robotics and Automation*, pages 1446–1451, 2010.

[21] F. Teichteil-Königsbuch and P. Fabiani. Autonomous search and rescue rotorcraft mission stochastic planning with generic dbns. In *IFIP AI*, pages 483–492, 2006.

[22] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *International Conference on obotics and Automation*, volume 2, pages 1023–1029, 2000.

[23] R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras, and H. Das. The claraty architecture for robotic autonomy. In *Aerospace Conference*, volume 1, pages 121–132, 2001.