Planification d'une mission d'observation par allocation de tâches hiérarchiques pour une équipe de robots hétérogènes

Hung Cao, Simon Lacroix, Félix Ingrand

Planification d'une mission d'observation par allocation de tâches hiérarchiques pour une équipe de robots hétérogènes *

Hung CAO Simon LACROIX Félix INGRAND

CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France
Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France

Résumé

Cet article présente un système d'allocation de tâche pour la réalisation d'une mission d'observation d'une zone à priori connue par un ensemble de robots. Le système entrelace les processus de décomposition et d'allocation de tâches, et raisonne explicitement sur des coûts de navigation et sur la faisabilité de tâches d'observation en exploitant des modèles de l'environnement. Il est basé sur une approche hiérarchique qui traite efficacement des missions couvrant de grandes zones. L'approche est illustrée sur la base de modèles réalistes d'un environnement et de robots.

Mots Clef

Systèmes multi-robots, planification de mission distribuée, allocation de tâches, processus d'enchères.

Abstract

This article presents a multi-robot task allocation system to achieve the observation of a given known zone. The system interleaves the decomposition and allocation processes, and explicitely reasons on navigation costs and visibility constraints by exploiting models of the environment. It relies on a hierarchical approach to efficiently process missions covering large areas. The approach is illustrated with realistic world and robot models.

Keywords

Multi-robot systems, distributed mission planning, auction-based task allocation.

1 Introduction

Nous nous intéressons à la mise en œuvre d'une flotte de robots hétérogènes pour des missions d'observation et de surveillance d'une zone donnée (*e.g.* pour détecter la présence d'intrus). La zone à observer est de type rural avec quelques bâtiments, et impose des contraintes de traversabilité et de visibilité. Les robots peuvent être aériens (UAVs [1]) ou terrestres (UGVs [2]) avec des capacités de déplacement et d'observation différentes, mais complémentaires.

État de l'art. De nombreux travaux ont été menés sur les algorithmes de planification de tâches pour les missions d'observation et d'exploration d'un groupe de robots.

Une des *approches centralisées* est de décomposer l'environnement selon une règle statique et d'affecter les sous-zones obtenues aux robots. Ainsi [Rekleitis et al., 2004] alloue au groupe de robots un ensemble de sous-zones obtenu par une décomposition de type boustrophédon, ou [Mazza and Ollero, 2004] inverse le processus en partant d'une décomposition de la zone à observer en sous-zones auxquelles sont associées un balayage. Les arbres de recouvrement sont aussi utilisés comme règle de décomposition statique : [Elmaliach et al., 2007] construit d'abord l'arbre de recouvrement de la zone à observer, et divise ensuite cet arbre en plusieurs portions qui sont affectées aux robots. Cette division / affectation tend à minimiser le chemin parcouru de chaque robot, et la structure de l'arbre à recouvrement assure une couverture complète de la zone.

Parmi les approches distribuées, la plus populaire est l'allocation de tâche par enchères. Cette approche est efficace pour les missions trivialement décomposables en tâches, *e.g.* routage multi-robot, où les robots gagnent des tâches via des sessions d'enchères. [Lagoudakis et al., 2005] donne une description et une bonne analyse de ces algorithmes utilisés pour le problème de routage multi-robot où un ensemble de points de passage sont donnés en entrée. Ces algorithmes manipulent les tâches indépendantes et la description de la mission revient à un séquencement de ces tâches – mais ces hypothèses de séquentialité et d'indépendance ne conviennent pas à la description de missions plus sophistiquées.

Pour pallier cette insuffisance, les systèmes *M+* [Botelho and Alami, 1999] et *DEMiR-CF* [Sariel et al., 2007] considèrent des plans partiellement ordonnés. À tout moment, les enchères portent sur le sous-ensemble des tâches restantes à exécuter : le plan global de la mission est alloué de manière incrémentale au groupe de robots durant l'exécution via un processus itératif d'allocation. Un formalisme hiérarchique de tâche a été aussi proposé dans les travaux de [Mosteo and Montano, 2006], de [Gerkey and Matarić, 2002], et de [Zlot and Stentz, 2005] :

*Ces travaux ont été partiellement financés par la DGA (PEA Action) http://action.onera.fr/

1. Unmanned Aerial Vehicle
2. Unmanned Ground Vehicle

la structure hiérarchique y est initialement spécifiée avec la mission. Nous allons voir que ce n'est pas le cas dans notre approche où les robots sont hétérogènes, et où la décomposition de la mission dépend donc de l'affectation.

Approche. Notre approche s'appuie sur des modèles de l'environnement permettant d'expliciter les contraintes de traversabilité et de visibilité. Ces modèles reposent sur une représentation hiérarchique de type *quadtree*. Ils fournissent aux robots les coûts de traversabilité des zones, et les possibilités et coûts d'observation de chaque zone. L'objectif global de la mission est d'observer l'ensemble des zones à moindre coût. La répartition des tâches d'observation entre les robots participants se fait par enchères : chaque robot devant observer une zone peut la décomposer en *quadtree* et mettre aux enchères ces tâches résultantes. Les autres robots répondront à cette proposition en fonction de leurs capacités propres et de leur charge de travail déjà acquise. Cette algorithmique distribuée et récursive est exploitée *hors-ligne*, dans une phase préalable de planification de mission, et spécifie ainsi à chaque robot une mission "nominale", mais elle est aussi exploitée en ligne, lors de l'exécution si un aléa ou un échec le requiert.

Structure de l'article. La section suivante introduit les différents modèles exploités pour mener les processus d'affectation et de décomposition : modèles de l'environnement, modèles des actions de déplacement et de perception des robots, modèles de tâches et de plans. La section 3 décrit le processus d'enchères mis en œuvre, en détaillant les différents algorithmes nécessaires. La section 4 analyse quelques résultats, et enfin une discussion conclut l'article.

2 Formulation du problème

L'ensemble $\mathcal{R}$ des robots évoluent dans le même environnement Z. Les modèles de l'environnement construits et utilisés par les robots reposent sur une librairie qui permet de définir des graphes sur la base de modèles *raster* : les modèles que nous exploitons sont un modèle de traversabilité pour les tâches de déplacement, et un modèle numérique de terrain pour les calculs de visibilité. Les robots s'appuient sur cette librairie pour construire et négocier des plans, en évaluant le coût et l'utilité d'exécution de tâches (déplacements et observations) pour maximiser le profit des plans produits (ratio utilité/coût).

2.1 Modèles de l'environnement

Hiérarchisation en quadtree. L'environnement étant grand (de l'ordre d'un km^2), il est décomposé en plusieurs niveaux en *quadtree* (figure 1). Chaque robot, selon la

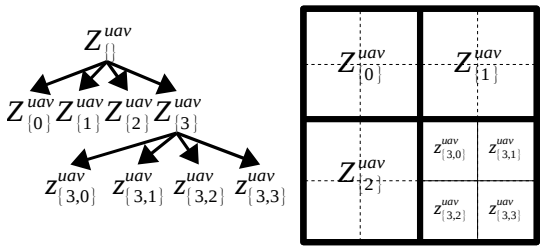

FIGURE 1 – Décomposition géométrique (droite), le *quadtree* correspondant (gauche). Chaque zone Z est identifiée par un vecteur d'indices I. Le vecteur d'indices vide I=() représente le noeud correspondant à la zone globale de la mission, et la dimension $dim(I)$ donne le niveau du noeud dans l'arbre.

portée de perception de son capteur, a un niveau maximal de décomposition df_p. L'opérateur de décomposition de zone Φ_Z associe à une zone abstraite Z_I une zone atomique z_I (aussi appelée *cellule*) si le niveau de décomposition dépasse la valeur df_p, et divise Z_I en quatre sous-zones abstraites $Z_{I\oplus i}$ sinon :

$$\Phi_Z(Z_I) = \begin{cases} \{z_I\} & \text{si } dim(I) == df_p \\ \{Z_{I\oplus 0},\ Z_{I\oplus 1},\ Z_{I\oplus 2},\ Z_{I\oplus 3}\} & \text{si } dim(I) < df_p \end{cases}$$

Modèle de déplacement. Le coût de déplacement dans une cellule est le produit de la distance euclidienne parcourue par le coût unitaire de traversabilité de cette cellule. Le modèle de traversabilité indique la classe c de chaque cellule, par exemple parmi l'ensemble *{plat, pente, accidenté, obstacle}* : à chacune de ces classes est associé un coût unitaire c_c (le coût associé à une cellule obstacle est naturellement infini). La carte de traversabilité est initialement décomposée en une structure de *quadtree* (figure 2), et le modèle permet de calculer le coût de déplacement $c_d(p_i, p_j)$ entre deux positions p_i et p_j quelconques.

Modèle de perception. Grâce à des calculs de visibilité établis sur la base du modèle numérique de terrain, les éléments suivants peuvent être évalués :

– Calcul de visibilité : $vs(p_i) = \{z_{I_0}, .., z_{I_m}\}$ est l'ensemble de zones atomiques z visibles de la position p_j. Pour une zone Z_I donnée, $\rho(Z_I \mid p_j)$ est le nombre de zones atomiques z appartenant à Z_I qui sont observées à partir de la position p_j : cela représente *l'utilité* ρ de la tâche

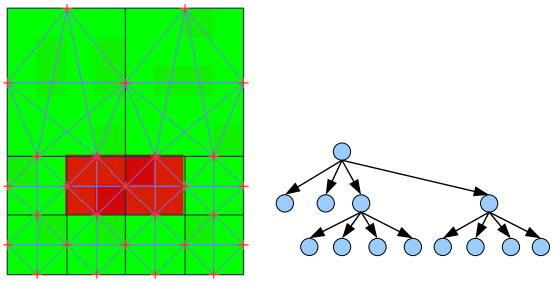

FIGURE 2 – Graphe de navigation construit sur une décomposition en *quadtree*, et *quadtree* correspondant.

d'observation de la zone Z_I depuis la position p_j :

$$\rho(Z_I \mid p_j) = card(vs(p_j) \cap Z_I) \quad (1)$$
$$= \sum_{i=0}^{3} \rho(Z_{I\oplus i} \mid p_j) \quad (2)$$

- $n\_best(Z_I)$ est l'ensemble des n meilleures positions d'observation de la zone Z_I (ensemble des positions p_j qui maximisent l'utilité $\rho(Z_i|p_j)$).
- $p_j^* = obs(Z_I \mid p)$ est la position d'observation qui maximise l'utilité ρ d'observation de la zone Z_I tout en minimisant le coût d'exécution du déplacement d'une position donnée p à la position p_j^*. p_j^* est déterminé en maximisant le profit $\delta(Z_I \mid p_j^*, p)$:

$$p_j^* = \underset{p_j \in n\_best(Z_I)}{\operatorname{argmax}} \delta(Z_I \mid p_j, p)$$

où

$$\delta(Z_I \mid p_j, p) = \omega_u * \rho(Z_I \mid p_j) - \omega_c * c(p, p_j)$$

Les pondérations ω_u (pour l'utilité) et ω_c (pour le coût) déterminent le comportement du système de robot dans la planification de la mission. Ainsi $\frac{\omega_u}{\omega_c} \longrightarrow \infty$ implique la production de plans qui maximisent le nombre de zones atomiques observées en ignorant le coût de déplacement et donc le temps de la mission, et $\frac{\omega_u}{\omega_c} \longrightarrow 0$ produit un comportement inverse.

- $p_P^* = obs(Z_I \mid P)$ est une extension de p_j^* sur le plan P : il s'agit de la meilleure position d'observation pour la zone Z_I étant donné le plan courant P du robot.

$$p_P^* = obs(Z_I \mid P) = \underset{p_i \in P}{\operatorname{argmax}} \delta(Z_I \mid obs(Z_I \mid p_i))$$

2.2 Modèles de tâches et de plans

Les robots sont capables d'effectuer deux actions : le déplacement $T_d(p_i, p_j)$ de p_i à p_j, et l'observation $T_o(Z_I, p)$ d'une zone Z_I à partir de la position p. La combinaison de ces deux actions définit la tâche $T_{do}(Z_I)$, qui consiste pour le robot à se déplacer à la position p^* pour observer la zone Z_I. Cette position p^* est donnée par $obs(Z_I \mid P)$, P étant le plan courant du robot.
Le plan du robot $P = \{\mathcal{T}, \Pi\}$ est une structure arborescente $\mathcal{T}$ dont les noeuds sont les tâches T_{do}. Il y a bijection entre un noeud de $\mathcal{T}$ et une zone Z_I dans le *quadtree* des zones. Par analogie avec l'opérateur Φ_Z de décomposition de zone, on définit l'opérateur de décomposition de tâche Φ_P :

$$\Phi_P(T_{do}(Z_I)) = \begin{cases} \{T_d(p_i, p^*) \oplus T_o(Z_I)\} & \text{si } \frac{\rho(Z_I|p^*)}{card(Z_I)} > \tau \\ \{T_{do}(Z_{I\oplus i})\}_{i\in 0..3} & \text{si } \frac{\rho(Z_I|p^*)}{card(Z_I)} \leq \tau \end{cases}$$

À côté de cette structure hiérarchique qui contient l'information sur les zones à observer Z_I, il existe une projection Π des tâches feuilles permettant de préciser leur ordre d'exécution et les différentes positions d'observation p_P^*.

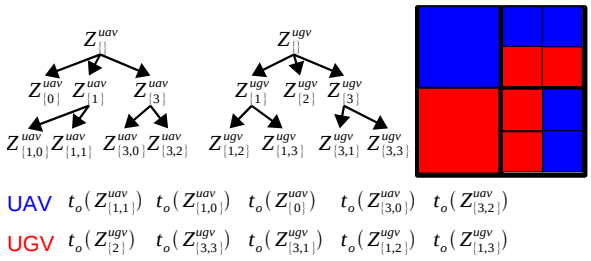

FIGURE 3 – Plans de deux robots UGV et UAV à un instant donné : structure hiérarchique $\mathcal{T}$ et projection Π séquentielle des tâches *exécutables* (feuilles de l'arbre).

2.3 Modèles de coût, d'utilité et de profit

Le coût c_d de l'action de déplacement $T_d(p_i, p_j)$ est donné par $c(p_i, p_j)$ défini par le *modèle de déplacement*, tandis que le coût c_p de l'action d'observation de $T_o(Z_I, p)$ est une constante k. Le coût de la tâche T_{do} est la somme des coûts de deux tâches T_d and T_o constituant T_{do}. L'utilité ρ de la tâche T_{do} se ramène à celle de la tâche d'observation T_o (une tâche de déplacement n'a pas d'utilité en soi) :

$$\rho(T_{do}(Z_I)) = \rho(T_o(Z_I)) = \rho(Z_I \mid p^*)$$

Pour calculer le coût d'une tâche correspondant à un noeud *non terminal*, il suffit de sommer le coût de ses fils :

$$\rho(T_o(Z_I)) = \sum_{t \in \Phi_P(T_o(Z_I))} \rho(t) \quad (3)$$

$$= \sum_{i \in 0..3} \rho(T_o(Z_{I \oplus i})) \quad (4)$$

Les coûts d'observation étant constants pour tout robot, le coût d'un plan P est la somme des coûts de déplacements, et son profit $\delta(P)$ est la somme des profits de ses tâches :

$$c(P) = \sum c(T_d(p_i, p_j)) = \sum c_d(p_i, p_j)$$

$$\delta(P) = \sum \delta(t_k) = \sum \delta(Z_I \mid p_i, p_j)$$

$\mathcal{P}$ étant l'ensemble des plans des robots, $P_i = \{\mathcal{T}, \Pi_i = [t_0^i, ..., t_{m_i}^i]\}$ le plan du robot $r_i \in \mathcal{R}$, la fonction à maximiser est la somme des profits de ces plans :

$$\sum_{P_i \in \mathcal{P}} \delta(P_i)$$

Par conséquent, le plan global optimal $\mathcal{P}^*$ est :

$$\mathcal{P}^* = \underset{\mathcal{P}}{\operatorname{argmax}} \sum_{P_i \in \mathcal{P}} \delta(P_i)$$

3 Construction du plan par enchères

3.1 Principe général

La planification de tâches multi-robot est souvent abordée en deux étapes : la première étape décompose la tâche en un ensemble de tâches élémentaires, que le système alloue aux robots lors de la deuxième étape. Mais une décomposition initiale biaise l'allocation, et inversement une allocation initiale force la décomposition des tâches par les robots auxquelles elles ont été affectées. Ceci n'est pas satisfaisant lorsque les robots sont hétérogènes : notre approche est basée sur un entrelacement de ces deux phases en une boucle de *décomposition-affectation*. Chaque tâche t de type T_{do} peut être décomposée par l'opérateur Φ_P en un ensemble de sous tâches abstraites ou élémentaires, et l'affectation est assurée par un processus d'allocation par enchères. Ces deux processus sont entrelacés jusqu'à ce qu'un *critère d'arrêt* Ψ soit satisfait (toutes les tâches sont élémentaires ou le gradient de l'évolution du profit total est inférieur à un seuil donné).

La boucle principale de planification sur chaque robot est décrite dans l'algorithme 1. Dans cette boucle, chaque robot choisit dans son plan courant une tâche à (re)distribuer via la création d'une session d'enchères après un accès exclusif au *token* d'allocation, qui assure qu'il n'y a qu'une seule session d'enchère en cours dans le système. Parallèlement à cette boucle, chaque robot est à l'écoute des propositions de participation aux sessions d'enchères. À chaque demande, un robot crée une session en tant que participant et exécute les opérations décrites dans l'algorithme 2 : il décompose d'abord la tâche, puis calcule son profit δ après insertion (algorithme 4) dans son plan courant. Il émet ensuite cette valeur d'enchère au vendeur. Le vendeur collecte l'ensemble β de ces valeurs δ et détermine l'affectation Γ^* de cette tâche aux meilleurs enchérisseurs (algorithme 5). À la fin de chaque session, chaque robot met à jour son plan **updatePlan** à partir de la structure d'affectation résultante Γ^*.

Algorithme 1: planProcess

Data : Plan vide $P_i = \{\mathcal{T}_i, \Pi_i\}$
Result : Plan final $P_i = \{\mathcal{T}_i^*, \Pi_i^*\}$

```
1  begin
2      while not Ψ(P_i) do
3          t = T_do(Z_I) ← selectTaskToAuction(P_i);
4          getNetworkToken;
5          Σ ← launchAuctionSession(T_do(Z_I));
6          releaseNetworkToken;
7          Γ* ← auctionResult(Σ);
8          updatePlan(P_i, Γ*);
9      end
10 end
```

Algorithme 2: bidder

Data : Plan du robot P_i, session d'enchère Σ
Result : vecteur des valeurs d'enchère $\beta(\Sigma)$

```
1  begin
2      t = T_do(Z_I) ← getTaskInAuction(Σ);
3      ST ← getChildrenNodes(t) ∪ {t};
4      foreach st ∈ ST do
5          β(Σ) ← β(Σ)∪ evalTaskIntoPlan(st);
6      end
7      emitBids(β(Σ))
8  end
```

3.2 Décomposition de tâches

L'opérateur Φ_P décompose $T_{do}(Z_I)$ en 4 sous tâches si elle est abstraite et en un ensemble vide si elle est élémentaire. Une tâche d'observation $T_{do}(Z_I)$ est élémentaire si

Algorithme 3: auctioneer

Data : session d'enchère Σ
Result : liste des affectations Γ

```
1  begin
2      Γ ← {};
3      t=T_do(Z_I) ← getTaskInAuction(Σ);
4      wD(t,Γ);
5      sendWinnerStructure(Γ);
6  end
```

la zone Z_I peut être totalement observée par le robot propriétaire de la tâche à partir d'une position accessible p^* — ce qui est détecté quand le ratio des cellules observées de la zone Z_I dépasse un seuil τ donné $\frac{\rho(Z_I|p^*)}{card(Z_I)} > \tau$.

3.3 Calcul de la valeur d'enchère

La valeur d'une enchère correspond à l'intérêt du robot à s'approprier une tâche, elle est déduite du calcul du profit du plan du robot avant et après l'insertion de cette tâche.

Algorithme 4: insertTask

Data : Plan courant P_i, tâche à insérer $t = T_{do}(Z_I)$
Result : Le profit additionnel de cette insertion $\delta(Z_I \mid P_i)$

```
1  begin
2      Π* ← insertionScope(Π);
3      foreach <t_prev,t_succ> ∈ Π* do
4          p* ← obs(Z_I | p(t_prev)) ;
5          p*_succ ← obs(Z_J | p*) ;
6          if δ > δ* then
7              δ* ← δ;
8              <t*_prev,t*_succ> ← <t_prev,t_succ>;
9          end
10     end
11     p(t) ← p*;
12     Π ← insert(Π, <t*_prev,t*_succ>, t) ;
13     δ(Z_I | P_i) ← δ_*
14 end
```

Pour calculer le profit $\delta(P \cup T_{do}(Z_I))$, il faut insérer la tâche dans le plan. Il faut d'abord l'insérer dans la structure hiérarchique (arbre de tâche) puis calculer la projection Π de ses feuilles. Le calcul de la projection Π revient à résoudre un TSP[3]. Pour une solution exacte, nous adoptons l'algorithme de CONCORDE [Applegate et al., 2001],

3. Travelling Saleman Problem

Algorithme 5: Winner determination algorithm

Data : Tâche en enchère $t = T_{do}(Z_I)$, liste des affectations Γ
Result : liste des affectations Γ

```
1  begin
2      β ← bids(Σ);
3      Γ^children ← { } ; tvalue ← 0;
4      foreach c ∈ Φ_P(t) do
5          Γ^children ← Γ^children ∪ wD(c, Γ);
6          tvalue ← tvalue + Γ^children(t)
7      end
8      <r,v> ←  argmax_{<r,v>∈β(t)}  v
9      if v > tvalue then
10         Γ ← Γ ∪ {< a, t >}
11     end
12     else
13         Γ ← Γ ∪ Γ^children
14     end
15 end
```

pour une solution rapide, nous utilisons l'approche suivante : pour chaque tâche, l'ensemble de tâches *voisines* Π^* dans la projection courante est déterminé, et le point d'insertion est celui qui maximise le profit additionnel (algorithme 4).
Le *scope* Π^* peut être la projection entière mais la structure hiérarchique nous permet d'avoir une heuristique pour réduire sa taille. En effet, une tâche $T_{do}(Z_I)$ est voisine à une tâche $T_{do}(Z_J)$ si les deux zones Z_I et Z_j sont adjacentes.

Algorithme 6: evalTaskIntoPlan

Data : Plan courant P_i, t=$T_{do}(Z_I)$
Result : Le profit additionnel $\delta(Z_I \mid P_i)$

```
1  begin
2      δ ← insertTask(t);
3      removeTask(t);
4      return δ
5  end
```

3.4 Détermination des gagnants

Pour chaque noeud dans l'arbre de tâches $T_{do}(Z_I)$ mis aux enchères, il y a un ensemble de valeurs d'enchères β associées à chaque robot participant à la session. Pour déterminer le gagnant, il faut trouver la partition de l'arbre maximisant le profit total. Puisque le profit d'un noeud $T_{do}(Z_I)$

peut s'exprimer avec une équation récursive, l'affectation optimale des noeuds de l'arbre peut être trouvée avec un algorithme de programmation dynamique (algorithme 5).

$$\delta(t = T_{do}(Z_I)) = \max(\max(\beta(t)), \sum_{st \in \Phi_P(t)} \delta(st))$$

4 Implémentation et résultats

Cette section présente l'architecture logicielle qui supporte les algorithmes développés, puis quelques analyses qui permettent d'évaluer l'impact de l'approche et l'influence de certains choix. Une comparaison avec une approche d'allocation classique sans hiérarchisation des tâches permet de démontrer l'utilité de la hiérarchisation pour la réduction de la complexité (nombre de sessions d'enchères), tout en gardant une efficacité proche de l'approche classique.

4.1 Implémentation

Une architecture logicielle ACT2A a été développée en C++, dans le but de valider et évaluer les algorithmes proposés, et de les intégrer à bord d'un système robotique réel. ACT2A a été conçu de manière à ce qu'une instance fonctionne sur chacun des robots. Ces instances communiquent en utilisant le *middleware* de communication YARP [Fitzpatrick et al., 2008].
Il s'agit aussi d'une architecture modulaire encapsulant plusieurs modules (figure 4) dédiés aux diverses fonctions telles que la communication, la gestion des sessions d'enchères, ou encore la gestion du *token* qui assure l'unicité de ces sessions. Ces modules sont des processus indépendants qui sont interruptibles puisqu'ils sont implémentés avec des machines à états capables de gérer des événements.

4.2 Résultats

Protocole d'expérimentation. Le système multi-robot est composé de trois robots évoluant sur un environnement de taille 100x100m de résolution 1m, un jeu de 10 modèles d'environnement est exploité (figure 5). Pour chaque expérimentation, le système de planification est lancé pour plusieurs valeurs de VIEWLENGTH (allant de 5m à 20m), qui représente la portée du capteur de perception de chaque robot. Les résultats finaux sont les moyennes des 10 valeurs obtenues pour ces 10 modèles.

Utilisation de CONCORDE. Nous étudions ici l'influence d'utilisation de CONCORDE dans le calcul de la trajectoire optimale (ou encore la projection Π de meilleur profit) passant par les positions p^* d'observation des zones dans le plan P. Il y a trois manières d'exploiter CONCORDE :

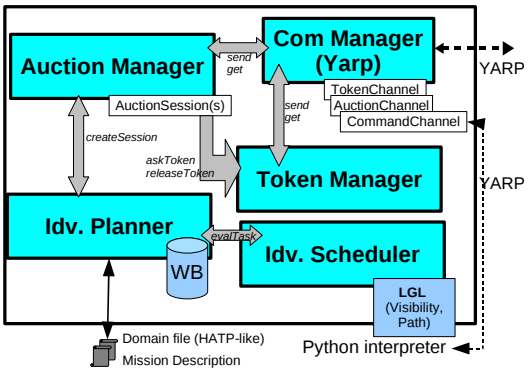

FIGURE 4 – Schéma des différents modules de l'architecture implémentée ACT2A.

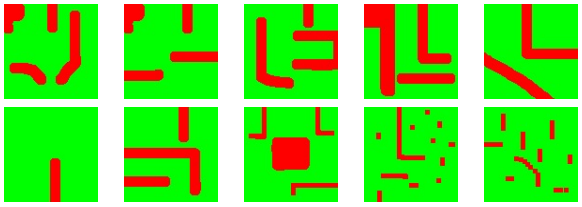

FIGURE 5 – Modèles d'environnements exploités pour les évaluations. Deux classes de traversabilité sont considérées : plan (vert) et obstacle (rouge).

1. INSERTION : La trajectoire est calculée de manière incrémentale par l'algorithme 4. La tâche $T_{do}(Z_I)$ est insérée à l'endroit qui maximise le profit additionnel.
2. MIXTE : utilisation de l'algorithme INSERTION lors de la construction du plan, mais le système optimise les trajectoires finales à l'aide de CONCORDE.
3. OPT : La trajectoire *optimale* est calculée par CONCORDE après chaque insertion de tâche.

Les résultats présentés dans la figure 6 permet de constater que l'algorithme MIXTE est un bon compromis entre la qualité du plan profit et le temps de calcul.

Stratégies d'enchère. Trois stratégies sont possibles pour mener les enchères :

1. MAXMIN : l'enchérisseur retourne le profit total de son plan avec la tâche $T_{do}(Z_I)$: $\delta(P \cup T_{do}(Z_I))$.
2. MAXSUM : le robot retourne seulement le *profit additionnel* qu'engendre l'insertion de la tâche $T_{do(Z_I)}$ dans le plan courant du robot : $\delta(P \cup T_{do}(Z_I)) - \delta(P)$.

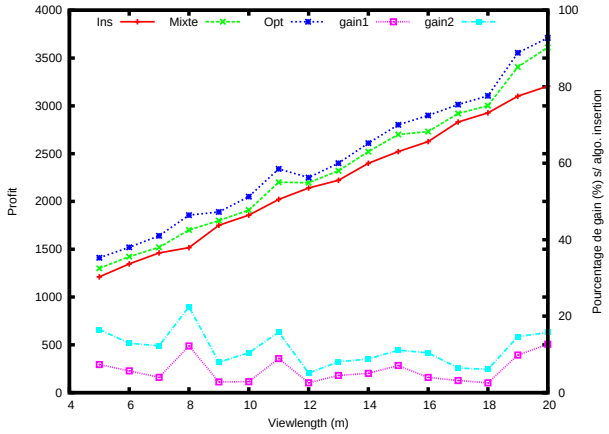

FIGURE 6 – Gain apporté par l'utilisation de CONCORDE dans le calcul de la trajectoire optimale. MIXTE donne une amélioration d'environ 10% sur INSERTION (GAIN1), et OPT d'environ 15% (GAIN2). La surcharge de calcul de OPT par rapport à INSERTION est d'environ 30%.

3. MAXMIX : stratégie mixte des deux stratégies MAXSUM et MAXMIN, la valeur d'enchère étant une combinaison linéaire du profit marginal et du profit total.

Les résultats obtenus (figure 7) montrent que les trois stratégies se comportent de manière similaire pour la valeur du profit total obtenu avec un petit avantage pour MAXSUM. Par contre, MAXSUM est très mauvaise pour la durée totale de la mission – qui est définie comme le temps d'exécution le plus long parmi ceux de tous les plans des robots. Cela n'est pas étonnant : en effet, en retournant le profit total du plan comme la valeur d'enchère, la stratégie MAXMIN tend à minimiser le temps de la mission tandis que MAXSUM, en retournant seulement le profit additionnel, maximise le profit global sans se préoccuper du temps de mission. Par conséquent, MAXSUM peut générer des plans déséquilibrés entre les robots. MAXMIN est une stratégie *collective* tandis que MAXSUM est plus *individualiste* : à chaque session, le plus fort emporte la tâche en jeu.

ACT2A vs Approche non hiérarchique. Nous voulons évaluer le gain de la hiérarchisation par rapport à une approche classique, où les tâches constituant la mission sont disponibles à l'avance et représentent des points de passage sur la zone à observer [Lagoudakis et al., 2005]. Pour obtenir cet ensemble de points de passage, nous utilisons une décomposition régulière de la zone à observer en une grille de cellules de taille 2∗VIEWLENGTH. Les résultats montrés figure 8 confirment l'intérêt de notre approche, qui

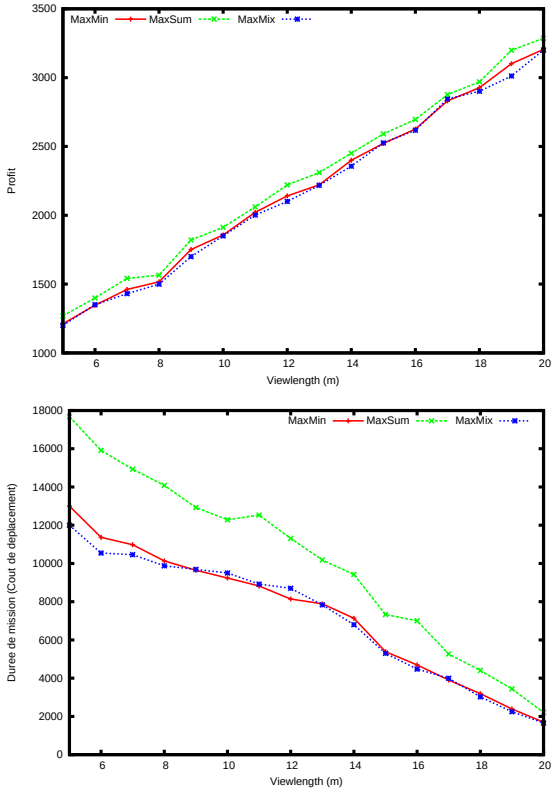

FIGURE 7 – Évaluation des stratégies d'enchère via le profit δ et la durée totale de la mission. MAXMIN et MAXMIX se comportent assez bien dans les deux cas tandis que MAXSUM donne une durée totale de mission d'environ 35% supérieure.

engendre environ 30% de sessions d'enchères en moins que l'approche classique tout en fournissant un profit proche (-3%) de celui obtenu par l'approche classique.

5 Bilan et perspectives

Nous avons présenté une approche distribuée de planification de mission d'observation pour un groupe de robots hétérogènes basée sur le paradigme d'allocation de tâche par enchères. Elle propose une hiérarchisation de la mission pour permettre d'une part de traiter des zones de grandes dimensions, et d'autre part d'entrelacer les processus de décomposition et d'allocation de tâches. Cet entrelacement permet de traiter les deux processus quasiment simultanément, réduisant ainsi l'influence de l'un sur l'autre.

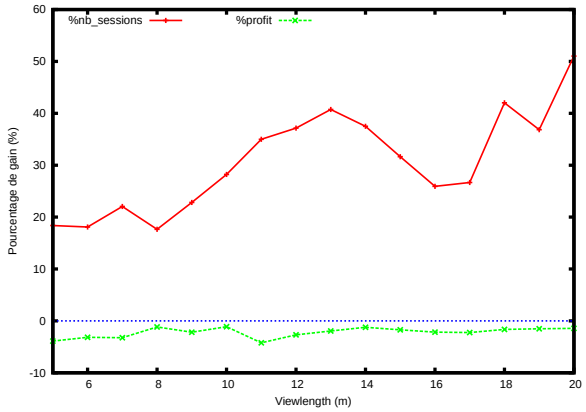

FIGURE 8 – Gain en profit et en nombre de sessions d'enchères par la hiérarchisation par rapport à l'approche classique d'allocation des points de passage. Notre approche donne des profits à peine inférieurs, mais permet de réduire considérablement le nombre de sessions d'enchères.

Nous avons mené plusieurs analyses pour comparer différents choix algorithmiques et évaluer l'intérêt de la hiérarchisation et de l'entrelacement des processus de décomposition et d'allocation par rapport à une approche classique du problème pour des missions d'observation. La hiérarchisation permet de réduire de 30% le nombre total de sessions d'enchères (et donc le nombre de messages échangés) avec un résultat proche de celui de l'approche classique.

Nos efforts portent maintenant sur l'extension de cette approche afin de considérer des contraintes de communication périodiques entre les robots et la station de base. Les travaux considérant les contraintes de communication dans un processus d'allocation [Mosteo et al., 2008, Atay, 2008] ne tolèrent pas de rupture de communications, ce qui est très contraignant : nous pensons pouvoir satisfaire des contraintes intermittentes de communications en les considérant lors de la projection d'un plan hiérarchique en tâches exécutables.

L'approche est implémentée au sein d'une architecture logicielle qui va permettre son intégration à bord de systèmes robotiques réels en vue d'expérimentations. En particulier, cette architecture va permettre de relancer un processus d'enchères lors d'aléas d'exécution, qui empêchent la réalisation plans initiaux (à cause d'information erronées dans les modèles de l'environnement par exemple). Lors de tels aléas, les plans peuvent être *réparés* par la mise en œuvre de nouvelles sessions d'enchères.

Références

[Applegate et al., 2001] Applegate, D., Bixby, R., Chvátal, V., and Cook, W. (2001). TSP cuts which do not conform to the template paradigm. In *Computational Combinatorial Optimization, Optimal or Provably Near-Optimal Solutions [based on a Spring School]*, pages 261–304, London, UK. Springer-Verlag.

[Atay, 2008] Atay, N. (2008). *Connectivity Maintenance and Task Allocation for Mobile Wireless Sensor Networ*. PhD thesis, Washington University in St. Louis.

[Botelho and Alami, 1999] Botelho, S. and Alami, R. (1999). M+ : a scheme for multi-robot cooperation through negotiated task allocation and achievement. *IEEE ICRA*, pages 1234–1239.

[Elmaliach et al., 2007] Elmaliach, Y., Agmon, N., and Kaminka, G. (2007). Multi-robot area patrol under frequency constraints. In *IEEE ICRA*.

[Fitzpatrick et al., 2008] Fitzpatrick, P., Metta, B., and Natale, L. (2008). Towards long-lived robot genes. *Robotics and Autonomous Systems*, 56 :29–45.

[Gerkey and Matarić, 2002] Gerkey, B. and Matarić, M. (2002). Sold ! : Auction methods for multi-robot coordination. *IEEE Transactions on Robotics and Automation*, 18(5) :758–768.

[Lagoudakis et al., 2005] Lagoudakis, M., Markakis, E., Kempe, D., Keskinocak, P., Kleywegt, A., Koenig, S., Tovey, C., Meyerson, A., and Jain, S. (2005). Auction-based multi-robot routing. In *Robotics : Science and Systems*, pages 343–350.

[Mazza and Ollero, 2004] Mazza, I. and Ollero, A. (2004). Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *7th International Symposium on Distributed Autonomous Robotic Systems, Toulouse (France)*.

[Mosteo and Montano, 2006] Mosteo, A. and Montano, L. (2006). Simulated annealing for multi-robot hierarchical task allocation with flexible constraints and objective functions. In *IROS'06 workshop on Network Robot Systems : Toward intelligent robotic systems integrated with environments*.

[Mosteo et al., 2008] Mosteo, A., Montano, L., and Lagoudakis, M. (2008). Multi-robot routing under limited communication range. *IEEE ICRA*, pages 1531–1536.

[Rekleitis et al., 2004] Rekleitis, I., Lee-Shue, V., New, A., and Choset, H. (2004). Limited communication, multi-robot team based coverage. In *IEEE ICRA*, pages 3462–3468.

[Sariel et al., 2007] Sariel, S., Balch, T., and Erdogan, N. (2007). Incremental multi-robot task selection for resource constrained and interrelated tasks. In *IROS*, pages 2314–2319.

[Zlot and Stentz, 2005] Zlot, R. M. and Stentz, A. (2005). Complex task allocation for multiple robots. In *IEEE ICRA*, pages 1515 – 1522.