THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par **l'Institut Supérieur de l'Aéronautique et de l'Espace**
Spécialité : Systèmes embarqués

Présentée et soutenue par **Grégory BEAUMET**
le 10 octobre 2008

Planification continue pour la conduite d'un satellite d'observation agile autonome

---

JURY

M. François Charpillet, président du jury
M. Brahim Chaib-draa, rapporteur
Mme Marie-Claire Charmeau
M. Jean-Pierre Merlet, rapporteur
M. Thierry Vidal
M. Gérard Verfaillie, directeur de thèse

École doctorale : **Systèmes**

Unité de recherche : **Équipe d'accueil ISAE-ONERA CSDV**

Directeur de thèse : **M. Gérard Verfaillie**

Planification continue pour la conduite d'un satellite d'observation agile autonome

Aujourd'hui la quasi-totalité des engins spatiaux sont entièrement contrôlés depuis le sol. Des plans d'activités précis sont téléchargés régulièrement du sol vers l'engin, auquel la part d'autonomie laissée est réduite au minimum : régulation thermique, contrôle d'attitude.
Dans le cadre d'un programme commun au CNES et à l'ONERA (programme AGATA : Architecture Générique pour l'Autonomie : Tests et Applications) portant sur l'autonomie des engins spatiaux, nous avons envisagé de déplacer les mécanismes de planification d'un satellite agile d'observation équipé d'un instrument de détection de la couverture nuageuse, du sol vers le bord, afin d'augmenter sa réactivité et son efficacité.
Nous avons adapté au cas d'un satellite agile et nous avons amélioré une architecture générique réactive-délibérative développée dans le cadre du projet AGATA. Cette architecture est centrée sur une tâche réactive capable de répondre instantanément aux stimuli de l'environnement (réception de nouvelles requêtes d'observation, état de la couverture nuageuse). Cette tâche réactive est équipée de règles de décision par défaut et contrôle une tâche délibérative qui s'exécute en parallèle et utilise au mieux le temps dont elle dispose pour choisir de façon optimale la prochaine décision à prendre.
Il devient alors réaliste de concevoir un satellite d'observation décidant de façon autonome et en temps-réel des actions à exécuter parmi des actions de manœuvre orbitale (lorsqu'il s'éloigne trop de son orbite de référence), de rendez-vous en attitude, de détection de la couverture nuageuse, d'observation de zones au sol, de télédéchargement de données, de rechargement des batteries et de pointage géocentrique.
Le développement d'un environnement de simulation temps-réel nous a permis d'évaluer le gain apporté par cette architecture, par les mécanismes de décision mis en place (règles de décision et algorithmes d'optimisation) et par la détection de la couverture nuageuse à bord du satellite.

Mots clés : Décision, Planification, Autonomie, Mécanique Spatiale, Observation de la Terre.

Continuous planning for the control of an autonomous and agile Earth-observing satellite

Most of the currently active spacecrafts are entirely controlled from the ground. Precise activity plans are regularly uploaded from the ground to the spacecraft for which autonomy consists in thermal regulation and attitude control.
Taking part in a joint CNES-ONERA project (AGATA: Autonomy Generic Architecture: Test and Applications) which aims at developing techniques for improving spacecraft autonomy, we worked on autonomous planning and decision-making mechanisms for an agile Earth-observing satellite equipped with a cloud cover detection instrument in order to increase spacecraft reactivity and efficiency.
We adapted to the case of an agile satellite and we improved a generic reactive-deliberative architecture developed in the context of the AGATA project. This architecture is centred on a reactive task able to react instantaneously to environment stimuli (arrival of new observation requests, cloud cover information). This task is equipped with reactive decision rules by default and controls a deliberative task running in parallel and using the available time to make the best next decision.
It becomes now feasible to design an observing satellite deciding autonomously and in real-time on the actions to be performed among orbital maneuver actions (when the drift of its orbital trajectory from the reference orbit becomes too important), attitude rendez-vous, cloud cover detection, observations of ground areas, data downloads, recharges of its batteries and geocentric pointings.
The development of a real-time simulation tool allowed us to evaluate the gain provided by this architecture, by the developed decision-making mechanisms (decision rules and optimization algorithms) and by the cloud cover detection onboard the satellite.

Keywords : Decision, Planning, Autonomy, Spatial Mechanics, Earth Observation.

Remerciements

Les travaux réalisés au cours de la présente thèse ont été cofinancés par le Centre National d'Études Spatiales et l'ONERA, et se sont inscrits dans le cadre d'un projet commun à ces deux organismes, portant sur l'autonomie des engins spatiaux.

Je remercie Patrick Fabiani et Jean-François Gabard de m'avoir accueilli dans l'unité de recherche Conduite et Décision du Département Commande des Systèmes et Dynamique du vol de l'ONERA.

Merci à tous les membres de ce département, et en particulier aux trois Michel qui ont contribué à l'avancée de mes travaux : merci à Michel Corrège que, malgrè toutes mes tentatives, je n'aurai jamais réussi à coller sur la gestion du matériel informatique dont je disposais ; merci à Michel Llibre pour le temps qu'il a pu me consacrer sur tous les aspects mécaniques du sujet, et pour sa contribution au légendaire débat Java Vs C ; enfin, merci à Michel Lemaître pour ses conseils et sa participation, avec Frédérick Garcia, Félix Ingrand et Thierry Vidal, au suivi annuel de mes travaux au travers des comités de thèse.

Je tiens à remercier tout particulièrement mes deux encadrants Marie-Claire Charmeau et Gérard Verfaillie pour m'avoir permis d'effectuer cette thèse dans d'excellentes conditions, pour m'avoir fait confiance et soutenu tout au long de ces trois années.

Un grand merci également à Solange Lemai-Chenevier et Antoine Flippo qui ont suivi de près mes travaux et dont les conseils m'ont été forts utiles.

Je remercie les membres de mon jury de thèse pour la qualité de leur analyse et la pertinence de leurs questions lors de la soutenance. Merci en particulier à François Charpillet d'avoir accepté de présider le jury.

Merci à Brahim Chaib-Draa et Jean-Pierre Merlet, rapporteurs du présent manuscrit, pour le temps qu'ils ont du consacrer à sa lecture, et pour leurs conseils avisés qui m'ont permis d'améliorer la qualité de sa rédaction.

Un grand merci à tous les doctorants que j'ai pu cotoyer au cours de ces trois années : Greg, Nico, Damien, Alberto, Sylvain, Clément, Cédric, Flo, Michel, Vincent, Julien, Patrice, Stéphane, Romain, Charles, Olivier, Florent, Vital, José et Laura, et qui ont contribué d'une manière ou d'une autre et parfois même inconsciemment, à la réussite de

ces travaux. Un merci particulier à St'f pour les longues pauses café que l'on a partagées et qui ont été, entre autres, une aide indéniable à la rédaction de ce mémoire.

Merci aussi aux footeux du mardi midi parmi lesquels : Matthieu, Domingo, Brice, Madjid, Bernhard, Laurent, Manu, Guillaume, Jean-Loup, Thierry, Julien, Charles, Alberto, ...

Je termine par une pensée à mes parents et mes sœurs que je ne remercierai sûrement jamais assez pour m'avoir soutenu sans relâche au cours de toutes ces années d'études.

Table des matières

Table des figures

Liste des tableaux

Table des algorithmes

Introduction

Contexte

Aujourd'hui, avec l'augmentation croissante de la complexité des systèmes spatiaux, l'autonomie à bord de tels engins apparaît comme une nécessité. En plus de permettre à l'engin d'agir en l'absence de communication avec les stations terrestres, l'automatisation des activités de commande - contrôle permet de diminuer les coûts de maintenance, et d'alléger la charge de travail des opérateurs au sol.

En particulier, le contexte d'une mission d'observation de la Terre par un satellite capable de détecter en avant la couverture nuageuse, requiert une forte autonomie décisionnelle embarquée. Que ce soit pour réagir à l'arrivée de nouvelles informations sur la couverture nuageuse, à des évènements extérieurs inattendus ou à des défaillances de certains de ses instruments, l'autonomie permet à un satellite de s'affranchir des délais habituels d'une boucle de programmation passant par le sol, souvent incompatibles avec les temps de réaction demandés. C'est ainsi que le satellite Earth Observing One [Chien *et al.*, 2004] de la NASA a fonctionné plusieurs mois avec des capacités de reconnaissance de phénomènes et de planification des observations à bord.

Des modes de raisonnement autonome ont déjà été testés et simulés avec succès pour des satellites non agiles[1] d'observation de la Terre [Damiani, 2005]. L'objectif de la présente thèse, proposée conjointement par le CNES et l'ONERA, est d'étendre ce type de mécanismes au cas beaucoup plus complexe des satellites agiles de dernière génération.

Principales contributions

Cette étude s'est principalement intéressée aux mécanismes de prise de décision pour un satellite d'observation agile autonome. Elle a consisté dans un premier temps à modéliser

[1] Un satellite agile est un satellite capable de contrôler son orientation autour de son centre d'inertie.

dans le cadre CNT (Constraint Network on Timelines [Verfaillie *et al.*, 2009]), le problème de décision auquel est confronté un tel satellite.

Dans le cas particulier d'un satellite agile, une capacité de calcul de trajectoires en attitude à bord est indispensable à tout développement plus poussé de mécanismes autonomes à bord de l'engin. Nous avons donc proposé une approche d'estimation de ces trajectoires à bord du satellite, reposant sur la résolution d'un problème de satisfaction de contraintes par un solveur de CSP continu.

Ce problème une fois résolu, nous avons développé une architecture de contrôle pour le satellite, basée sur une architecture innovante proposée par [Verfaillie, 2006], que nous avons étendue, améliorée et adaptée au cas particulier d'un satellite agile.

Des mécanismes de décision réactifs et un algorithme d'optimisation de type glouton stochastique itéré ont été implémentés au sein de cette architecture de contrôle.

Cette architecture a ensuite été intégrée dans le logiciel de vol du satellite, lui-même intégré dans un environnement de simulation temps-réel que nous avons spécialement développé pour l'occasion.

Cet environnement nous a finalement permis de tester et d'évaluer le comportement des différents mécanismes de prise de décision que nous avons mis en place et de montrer l'intérêt d'embarquer des mécanismes autonomes à bord d'engins spatiaux.

Plan du manuscrit

Ce manuscrit suit l'organisation suivante :

Le **chapitre 1** définit diverses notions dans les domaines de l'autonomie et de la prise de décision. Il parcourt également un éventail non exhaustif de travaux qui ont été réalisés sur ces sujets, en particulier ceux qui nous ont été utiles au cours de cette étude.

Le **chapitre 2** décrit le contexte de l'étude, en présentant le projet AGATA (Architecture Générique d'Autonomie, Tests et Application) commun au CNES, au LAAS et l'ONERA, dont l'étude fait partie.

Le **chapitre 3** définit et modélise le problème de décision que nous avons tenté de résoudre au cours de cette étude.

Le **chapitre 4** s'intéresse aux aspects cinématiques du problème. Après une brève introduction de la mécanique spatiale, il décrit les méthodes existantes de contrôle d'attitude des satellites et présente une méthode de calcul de trajectoires en attitude à bord d'un satellite agile.

Munis de ces mécanismes de calcul de trajectoires en attitude, nous décrivons, dans le **chapitre 5**, le logiciel de vol complet développé pour le contrôle d'un satellite agile d'observation de la Terre.

Le **chapitre 6** décrit l'environnement de simulation temps-réel mis en place pour évaluer les mécanismes de décisions implémentés dans le logiciel de vol de l'engin. Il présente l'ensemble des modèles utilisés pour simuler le comportement du satellite et de son environnement, ainsi que les différentes fonctionnalités mises à disposition de l'utilisateur.

Enfin le **chapitre 7** présente les évaluations hors ligne et en ligne des mécanismes de décision mis en place.

Chapitre 1

État de l'art

1 Autonomie

Le degré d'autonomie d'un véhicule définit son aptitude à prendre des décisions sans intervention d'un opérateur. [Verfaillie, 2005] recense quatre grandes classes de besoins motivant le développement d'une plus grande autonomie des engins spatiaux :

1. des besoins de *simplification* et de *réduction du travail* réalisé dans les centres de contrôle (diminution des astreintes du personnel au sol) ;
2. des besoins de *contrôle* de phases plus ou moins *critiques* des missions, sur des périodes où la communication avec le sol n'est pas possible (les fenêtres de visibilité, et donc de communication, d'un satellite défilant avec une station sol sont rares et de courte durée) ou n'est pas compatible avec les délais de réaction nécessaires (sonde interplanétaire) ;
3. des besoins d'*augmentation de la disponibilité* des systèmes spatiaux, par l'existence à bord de mécanismes permettant une réaction immédiate à des défaillances et une reprise de l'activité normale, même sur des périodes où la communication avec le sol n'est pas possible (besoin de systèmes de FDIR[1] autonomes) ;
4. des besoins d'*amélioration du retour* des missions spatiales en termes de quantité et de qualité des données recueillies, par la présence à bord de mécanismes capables de prendre à tout moment et sans passer par le sol, des décisions adaptées à la situation et aux objectifs courants.

1.1 Les différents niveaux d'autonomie

Plusieurs classifications ont été proposées dans la littérature, pour organiser les différents niveaux d'autonomie. Une première classification, définie par Fargeon [Fargeon,

[1] FDIR : Fault Detection, Identification and Recovering.

1993] (tableau 1.1), utilise la localisation géographique de l'opérateur par rapport à l'engin comme critère de classification.

Niveau d'autonomie	Description
engin semi-autonome	l'opérateur est à bord de l'engin mais reçoit une aide à la décision
engin télécommandé	l'opérateur est déporté mais en vue directe avec l'engin
engin téléopéré	l'opérateur est déporté et n'est pas en vue directe avec l'engin ; l'environnement lui est restitué (téléprésence)
engin autonome	l'engin dispose de capacités significatives de prises de décisions, de mobilité et d'adaptation à la mission

TAB. 1.1: Niveaux d'autonomie définis par Fargeon [Fargeon, 1993]

Dans le domaine spatial, la détermination du niveau d'autonomie bord pour une mission donnée fait apparaître une distinction claire entre les activités de l'engin en mode nominal, et les ses activités en présence d'anomalies. Ainsi, dans [Anadon *et al.*, 2005], les niveaux d'autonomie sont définis de la façon suivante :

- en fonctionnement nominal :
  - **Niveau 1N** : maintenir le point de fonctionnement d'un asservissement bord sur une consigne en provenance du sol (fonctionnement télécommandé) ; ce niveau n'est envisageable que si le satellite est toujours en visibilité d'un centre de contrôle, ce qui est par exemple le cas pour un satellite géostationnaire ;
  - **Niveau 2N** : exécuter un plan de travail[2] construit au sol, indépendamment de résultats ou d'évènements bords (fonctionnement automatique) ; c'est par exemple le cas des satellites défilants d'observation de la Terre, Spot ;
  - **Niveau 3N** : modifier ou créer un plan de travail, pour atteindre des objectifs définis par le sol, en fonction de résultats obtenus à bord ou d'évènements internes ou externes au satellite (fonctionnement autonome) ;
- en présence d'anomalies :
  - **Niveau 1A** : isoler l'anomalie, abandonner l'activité en cours et se replier en mode survie dans l'attente d'une intervention des opérateurs au sol ;
  - **Niveau 2A** : isoler l'anomalie, abandonner l'activité en cours et se replier dans un mode d'attente d'une intervention des opérateurs au sol ;
  - **Niveau 3A** : isoler l'anomalie et se reconfigurer afin de poursuivre l'activité en cours.

Définition 1 Dans le cadre de notre étude, un satellite autonome sera un satellite capable de fonctionner avec les niveaux d'autonomie maximaux en mode nominal et en présence d'anomalie.

[2]Un plan de travail se présente sous la forme d'une séquence d'actions à effectuer à des dates précises.

1.2 Les différentes architectures de systèmes autonomes

D'après [Hassoun, 1997], on peut distinguer trois classes d'architectures de systèmes autonomes :

- les architectures *réactives* qui ne comportent pas de niveau de planification. Le robot réagit uniquement aux stimuli externes de l'environnement ;
- les architectures *délibératives* qui comportent plusieurs niveaux décisionnels dont un niveau de planification des actions du système ;
- et les architectures *hybrides* qui combinent les avantages des deux approches précédentes.

1.2.1 Architectures réactives

Cette catégorie regroupe les architectures composées d'entités distinctes représentant chacune un comportement simple de l'engin. Il n'y a pas de forme de raisonnement centralisé, ni de modèle global du système et de l'environnement. Le comportement d'un tel agent n'est donc pas planifié, mais résulte des interactions entre ces entités et l'environnement qui fonctionnent tous de manière concurrente.

L'architecture de subsomption L'architecture réactive la plus connue est l'architecture de subsomption (Subsumption Architecture en anglais) définie dans [Brooks, 1986]. Elle définit une hiérarchie de niveaux associés chacun à une tâche ou à un comportement particulier de l'engin. Les niveaux supérieurs réalisent les tâches les plus abstraites et ont une priorité plus petite que les niveaux inférieurs correspondant aux tâches les plus concrètes et les plus simples. La réalisation d'une tâche prioritaire de bas niveau peut ainsi modifier l'entrée d'une tâche de niveau supérieur (*suppression*) et même invalider sa réalisation (*inhibition*).

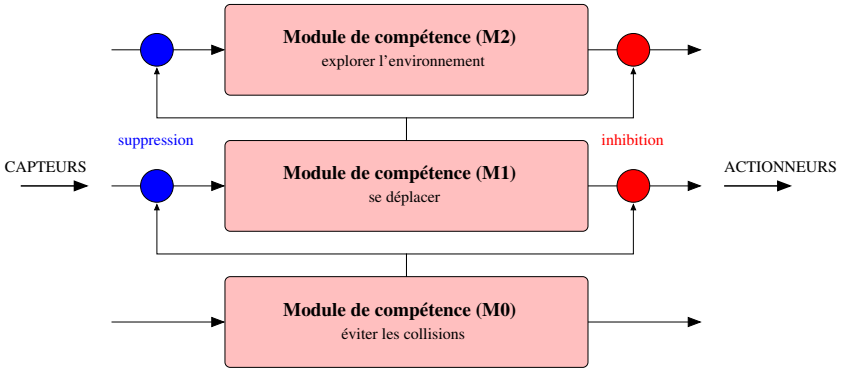


FIG. 1.1: Architecture de subsomption

La figure 1.1, tirée de [Chanthery, 2005], représente une telle architecture composée

de trois tâches correspondant à trois niveaux hiérarchiques $M0$, $M1$, $M2$, pour un robot mobile d'exploration planétaire :

- $M0$: éviter la collision avec d'autres objets ;
- $M1$: se déplacer dans l'environnement tout en évitant les obstacles ;
- $M2$: explorer l'environnement en utilisant les observations visuelles pour choisir les lieux intéressants et les déplacements du niveau $M1$.

1.2.2 Architectures délibératives

Dans ce type d'architecture, l'engin doit réaliser une tâche particulière, de manière autonome. Il est doté pour cela d'une fonction de délibération, appelée aussi planification. Aucune action de l'engin n'est exécutée si elle n'a pas été auparavant planifiée.

Pour palier au manque de réactivité d'une telle architecture, des composantes réactives y sont souvent intégrées. Ainsi, même si l'architecture est fortement délibérative, elle peut aussi être considérée comme hybride.

On présente dans cette section quelques-unes des architectures à composantes fortement délibératives, les plus connues.

L'architecture TCA (Task Control Architecture) Développée par Simmons [Simmons, 1994], cette architecture est organisée en modules agencés autour d'un contrôleur centralisé qui coordonne leurs interactions. Un but est décomposé en arbre de tâches contenant des sous-buts atteignables par les modules. Toute la communication est centralisée dans le contrôleur qui doit également répartir les tâches entre modules, en accord avec les contraintes temporelles imposées par le système.

L'architecture LAAS L'architecture LAAS [Alami *et al.*, 1998] a été conçue pour le contrôle de robots mobiles. Elle est organisée en trois niveaux (voir figure 1.2), soumis chacun à des contraintes temporelles différentes et exploitant chacun une représentation de données qui lui est propre :

1. Le niveau décisionnel décide des actions à mener. Celles-ci doivent être réalisées en fonction de la tâche à accomplir et du contexte d'exécution. Ce niveau produit des plans et supervise leur exécution qui se fait par des requêtes envoyées aux niveaux inférieurs de l'architecture, tout en restant réactif aux évènements qui peuvent remonter de façon asynchrone.
2. Le niveau exécutif joue le rôle d'interface entre le niveau décisionnel et le niveau fonctionnel et sa fonction principale est de garantir la cohérence du niveau fonctionnel et de prévenir tout conflit.
3. Le niveau fonctionnel regroupe l'ensemble des actions élémentaires que le robot est capable d'exécuter. Ces actions sont décrites dans des modules fonctionnels qui encapsulent les traitements associés aux services demandés par le niveau décisionnel.

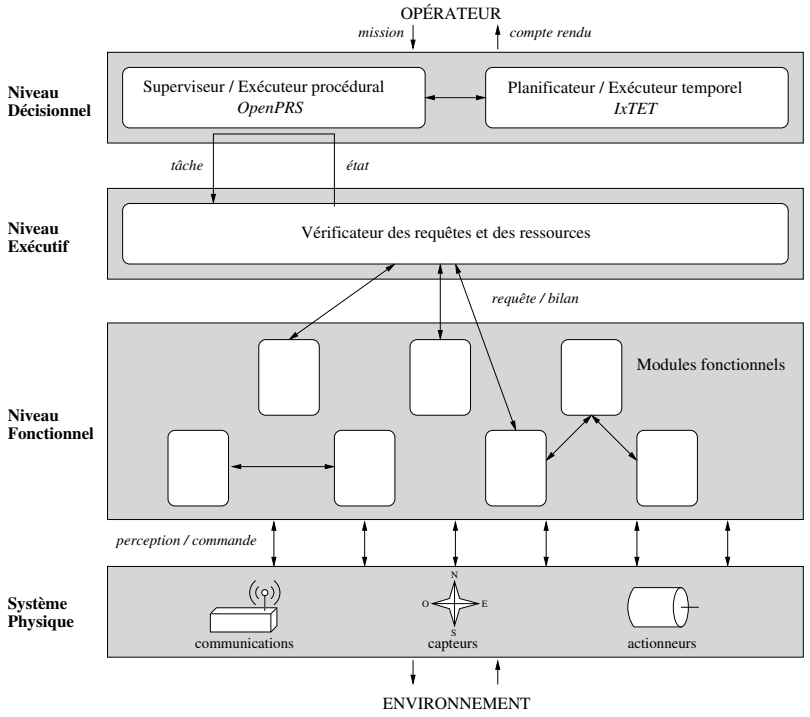


FIG. 1.2: Architecture LAAS

L'architecture du Remote Agent En 1999, la NASA a testé en vol, pendant quelques jours à bord de la sonde Deep Space One, un système autonome appelé *Remote Agent* [Muscettola *et al.*, 1998], dans le but de valider des fonctions de planification d'activités, de contrôle d'exécution de plan et de diagnostic et reconfiguration autonomes. L'architecture logicielle du Remote Agent regroupe trois fonctionnalités :

1. Un planificateur basé sur l'approche HSTS (*Heuristic Scheduling Testbed System*) [Muscettola, 1994]. Ce processus de planification n'est pas interruptible, et produit un plan solution flexible[3] sur un horizon donné (de l'ordre de la journée) au bout d'une durée de raisonnement assez longue (de l'ordre de l'heure).
2. Un exécutif dont la fonction principale est d'exécuter un plan en décomposant ses activités haut-niveau en primitives. Si une activité du plan échoue, l'exécutif peut essayer une méthode alternative cohérente avec la flexibilité temporelle du plan, ou peut demander la planification d'actions de reconfiguration au module MIR (voir [Pell *et al.*, 1998] pour plus de détails sur les interactions entre l'exécutif et le module MIR). En dernier recours, l'exécutif demande un nouveau plan au planificateur (voir [Pell *et al.*, 1997] pour plus de détails), et la sonde est placée dans un mode d'attente pendant la durée de planification.

[3]Les dates de début et de fin des activités ne sont pas fixées, mais bornées par un intervalle numérique

3. Un module de diagnostic, MIR (*Mode Identification and Recovery*), est lui-même décomposé en deux modules MI et MR permettant respectivement d'estimer l'état probable du système à partir des commandes envoyées par l'exécutif et des valeurs des capteurs, et de restaurer une fonctionnalité perdue en cas d'échec d'une activité (voir [Williams et Nayak, 1996] pour plus de détails).

Comme pour l'architecture LAAS, ces trois fonctionnalités sont organisées en trois couches (figure 1.3) et utilisent chacune une modélisation du problème qui lui est propre.

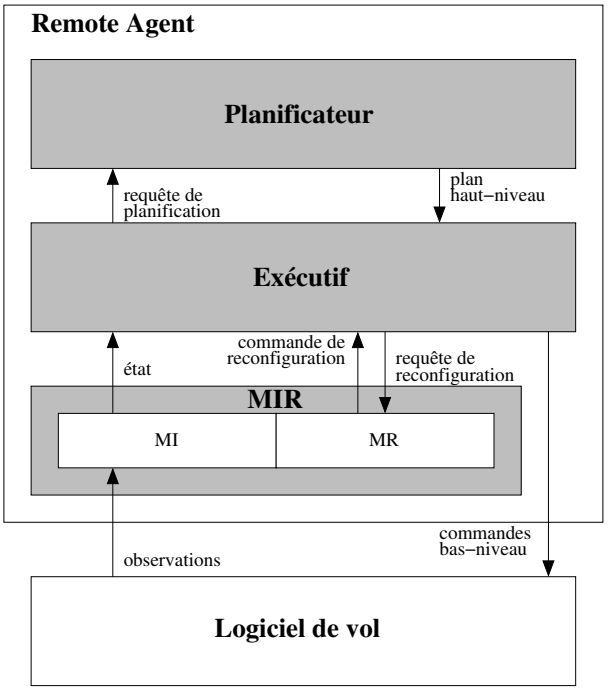


FIG. 1.3: Architecture du Remote Agent

L'architecture IDEA Pour palier à certaines difficultés rencontrées avec le Remote Agent (le fait que chaque module utilise ses propres mécanisme de raisonnement et modèle par exemple), la NASA a développée un nouveau concept d'architecture décisionnelle : IDEA (*Intelligent Distributed Execution Architecture*) [Muscettola *et al.*, 2002]. Cette architecture est centrée sur la planification. Un agent IDEA est composé de quatre éléments (voir figure 1.4). Le *plan database* représente les actions en cours d'exécution et les futures actions planifiées. Le *Plan runner* intègre les informations retournées par l'exécution et lance les prochaines actions du plan à exécuter. Le *model* contient les modèles associés aux actions activables par l'agent. Enfin, le *reactive planner* est activé par le *plan runner* lorsqu'une nouvelle information doit être prise en compte ou que l'action courante arrive à son terme.

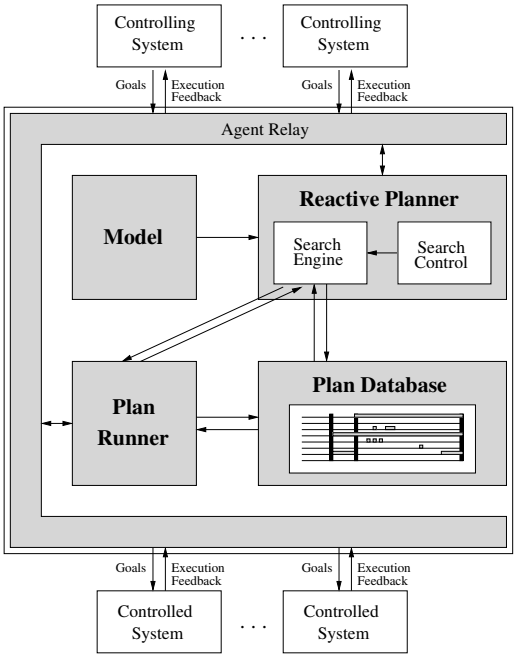


FIG. 1.4: Architecture d'un agent IDEA

1.2.3 Architectures hybrides

Une architecture hybride est composée d'une partie réactive et d'une partie délibérative disposant d'une représentation symbolique des connaissances et de capacités de raisonnement. À un comportement réactif de l'agent autonome aux changements de l'environnement vient donc s'ajouter un comportement pro-actif, dirigé par les buts.

L'architecture 3T et ATLANTIS Les premiers travaux dans cette voie ont donné naissance aux architectures 3T [Bonasso *et al.*, 1997] et ATLANTIS [Gat, 1992] qui sont très similaires. Elles sont organisées en trois niveaux hiérarchiques (voir figure 1.5) : un planificateur pour la partie délibérative (appelé *planning layer* pour 3T et *deliberator* pour ATLANTIS), un ensemble de mécanismes de commande réactifs (appelé *skill layer* pour 3T et *controller* pour ATLANTIS) et un niveau intermédiaire chargé de la séquence d'actions à exécuter (appelé *sequencing layer* pour 3T et *sequencer* pour ATLANTIS).

- La couche basse rassemble un ou plusieurs algorithmes implémentant des comportements réactifs simples liant étroitement capteurs et actionneurs.
- Le rôle du niveau intermédiaire est de sélectionner le ou les comportements qui doivent être actifs à un instant donné, et de fixer leurs paramètres éventuels. Pour 3T, ce niveau est implémenté en utilisant le langage RAP (Reactive Action Packages) proposé par R. J Firby [Firby, 1987], tandis qu'ATLANTIS s'appuie sur le langage ESL [Gat, 1997].

– La couche supérieure regroupe les tâches gourmandes en temps de calcul, comme les tâches de planification. Elle a pour but de fournir les séquences d'actions utilisées par la couche intermédiaire. Dans le cas de l'architecture 3T, la couche supérieure produit à l'avance le plan utilisé par la couche de séquencement, alors que dans le cas d'ATLANTIS, le plan est produit à la demande du niveau intermédiaire.

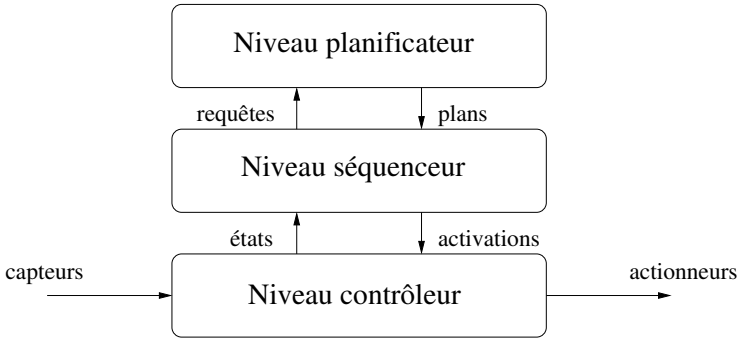


FIG. 1.5: Architecture à trois niveaux

L'architecture CLARAty L'architecture CLARAty (*C*oupled-*L*ayer *A*rchitecture for *R*obotic *Au*tonom*y*) a été développée a été développée conjointement par le Jet Propulsion Laboratory, le NASA Ames Research Center, l'Université Carnegie Mellon et l'Université du Minnesota [Estlin *et al.*, 2001].

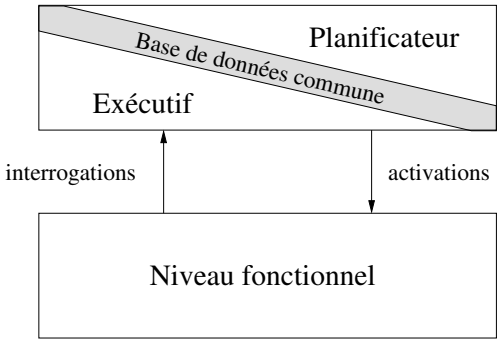


FIG. 1.6: Architecture CLARAty

Cette architecture est composée de deux niveaux présentés sur la figure 1.6 : un niveau fonctionnel, et un niveau décisionnel qui regroupe les niveaux exécutif et décisionnel d'une architecture trois niveaux classique. Ce niveau décisionnel unique a pour but d'éviter la gestion d'un modèle du système différent pour chaque niveau, et de permettre un meilleur contrôle des domaines décisionnels concurrents réactif/planification.

Le niveau fonctionnel est conçu suivant une approche objet, pour traduire la modularité de la couche matérielle et permettre une granularité d'abstraction au niveau décisionnel. Enfin, un objet du niveau fonctionnel peut utiliser un autre objet de ce niveau et l'état du système est mémorisé séparément dans chaque objet. Ces objets sont interrogés par le niveau décisionnel quand il a besoin d'une information.

L'architecture AGATA Cette architecture a été développée dans le cadre d'un projet commun au CNES, à l'ONERA et au LAAS. Elle repose sur une distinction claire entre les parties réactives et délibératives du système autonome : d'un côté une tâche décisionnelle réactive confère au système la capacité de réagir instantanément aux stimuli en provenance de l'environnement extérieur ; de l'autre côté, une ou plusieurs tâches décisionnelles délibératives sont en charge du calcul et de l'optimisation des décisions à prendre (tâches de planification et d'ordonnancement). Une description précise de cette architecture est faite au chapitre 2.

L'architecture ASE Testée depuis 2003 sur le satellite d'observation Earth Observing One [Chien *et al.*, 2004] de la Nasa, cette architecture confère au satellite des capacités de reconnaissance de phénomènes sol et de planification à bord, dans le but de démontrer l'utilité de l'autonomie décisionnelle embarquée.

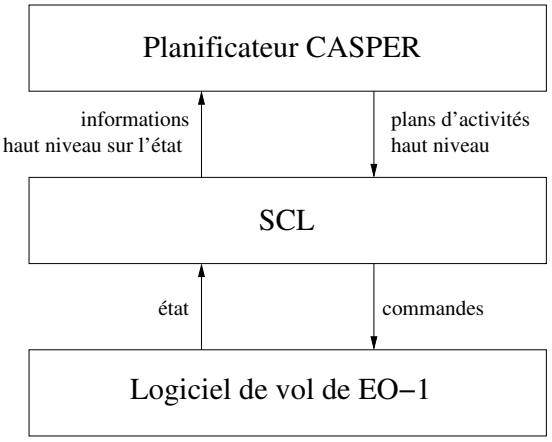


FIG. 1.7: Architecture ASE

Elle est classiquement organisée en trois niveaux (voir figure 1.7). Au plus haut niveau d'abstraction, un planificateur est en charge de la planification de la mission : le planificateur CASPER [Chien *et al.*, 2000b] ordonne les activités du satellite en respectant les contraintes sur les ressources de l'engin. Le système SCL (Spacecraft Command Language) génère alors la séquence détaillée des commandes correspondant aux activités planifiées. Au niveau inférieur, le logiciel de vol d'EO-1 est en charge du contrôle bas niveau du satellite.

2 Planification

2.1 Définitions

La planification s'est constituée très tôt comme une véritable discipline autonome au sein de la communauté Intelligence Artificielle, dont l'objet porte sur la construction automatique de plans d'actions [Ghallab *et al.*, 2004].

Définition 2 Planifier est une action de raisonnement qui consiste à sélectionner et organiser les actions futures d'un système contrôlé afin de remplir un certain objectif [Ghallab *et al.*, 2004].

Un objectif peut être ponctuel : amener le système dans un état final, ou étalé dans le temps : maintenir le système dans un certain état, assurer dans le temps un certain service ou optimiser un critère. Le but de la planification est de satisfaire complètement les objectifs (on parle alors de problèmes de satisfaction) ou de les satisfaire au mieux (on parle alors de problème d'optimisation). Selon [Smith *et al.*, 2000], la plupart des travaux de planification en Intelligence Artificielle peuvent se ranger dans l'une ou l'autre des trois catégories suivantes : planification classique, planification HTN (Hirearchical Task Network) et planification dans l'incertain.

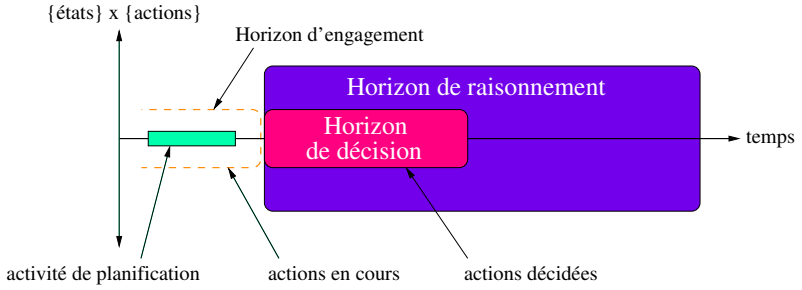


FIG. 1.8: Définition des paramètres d'une tâche de planification

Comme toutes les actions du système, la planification est une action qui prend du temps. Elle raisonne sur un sous-ensemble des activités possibles du système sur un certain intervalle temporel dans le futur : *l'horizon de raisonnement*. Ses résultats sont des décisions portant sur des actions du système que celui-ci peut engager sur un certain intervalle temporel dans le futur : *l'horizon de décision*. Enfin, en l'absence d'échec, les actions en cours d'exécution ne sont pas remises en question ; l'ensemble de ces activités sur lesquelles la tâche de planification ne revient plus constitue *l'horizon d'engagement* (voir figure 1.8).

Les relations entre ces différents horizons sont les suivantes :
- l'horizon d'engagement comprend l'action de planification ;
- l'horizon de décision débute immédiatement après l'horizon d'engagement ;
- l'horizon de décision est inclus dans l'horizon de raisonnement ;
- l'action de planification doit se terminer avant l'horizon de décision.

2.2 Approches de planification

Différentes approches, schématisées sur la figure 1.9, sont utilisées pour aborder un problème de planification :
- la planification prévisionnelle : construction d'un plan, puis exécution de ce plan par le niveau inférieur sans retour vers le niveau supérieur ;
- la planification prévisionnelle avec replanification : c'est une planification prévisionnelle avec la possibilité de relancer la planification en cas d'échec à l'exécution : pendant la replanification, le système physique est en attente ;
- la planification prévisionnelle conditionnelle : c'est une planification prévisionnelle où le plan produit anticipe d'éventuels évènements en prévoyant différents branchements ;
- la planification robuste : c'est aussi une planification prévisionnelle où un modèle des incertitudes est utilisé pour produire un plan résistant au mieux aux aléas ;
- la planification continue : le plan est constamment mis à jour au fur et à mesure de son exécution ;
- la définition d'une politique : ce n'est plus un plan mais une fonction qui, à tout état possible du système, associe l'action à réaliser.

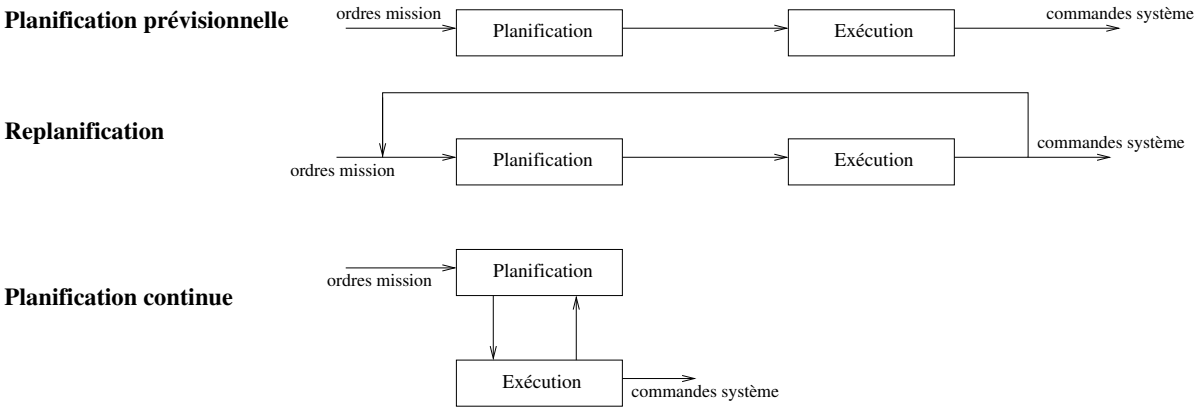


FIG. 1.9: Différentes approches pour décider et exécuter

2.3 Planification en ligne

Définition 3 On différencie la planification hors ligne de la planification en ligne :
- planifier hors ligne consiste à générer les décisions une fois pour toute et à les faire appliquer telles quelles par le contrôleur d'exécution du système ;
- planifier en ligne consiste à adapter les décisions en fonction des éventuels retours d'exécution.

Dans un environnement réel, dynamique et incertain, une seule action de planification est bien souvent insuffisante pour amener le système contrôlé dans l'état désiré. Il est alors nécessaire d'introduire des mécanismes d'interactions entre le planificateur et le contrôleur d'exécution du système.

Une première approche, intermédiaire, consiste à effectuer une partie du raisonnement hors ligne, puis à adapter si nécessaire le résultat en ligne. Dans [Khatib *et al.*, 2003], des plans d'observations sont élaborés hors ligne puis adaptés en ligne en fonction des consommations effectives de ressources et des nouvelles opportunités d'observations.

Une planification uniquement en ligne consiste à planifier sur un horizon de raisonnement de longueur arbitraire, à exécuter le plan résultant jusqu'à son terme ou jusqu'à ce que des aléas l'invalident (un niveau de ressource inattendu par exemple), puis à replanifier à partir de ce nouvel état (voir figure 1.10). Cette approche nécessite un contexte peu dynamique et plutôt déterministe, ce qui est souvent le cas dans le domaine spatiale, et en particulier dans le cas de la gestion d'une sonde spatiale en transfert interplanétaire. Le Remote Agent de la sonde Deep Space One [Muscettola *et al.*, 1998] utilisait cette approche avec une période de planification de quatre jours et une durée de planification de six heures, ce qui réduisait fortement sa réactivité.

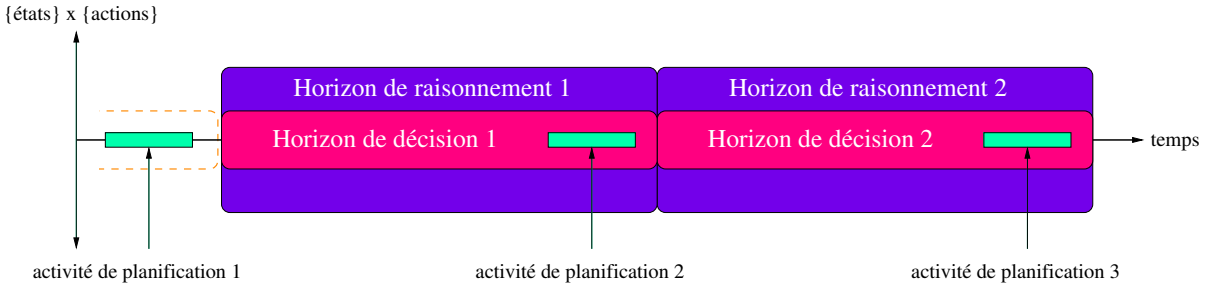


FIG. 1.10: Planification périodique

Une approche duale, dénommée planification continue, consiste en une unique action de planification permanente, maintenant à jour le plan d'activité du système. Les horizons de décision et de raisonnement sont glissants, c'est-à-dire qu'ils se décalent temporellement dans le futur, au fur et à mesure de la progression de la date courante. Leurs tailles sont

fixes et dépendent de la dynamique de l'environnement (voir figure 1.11). Le module de planification CASPER [Chien *et al.*, 2000b] de la NASA, embarqué à bord du satellite Earth Observing One [Chien *et al.*, 2004], utilise cette approche.

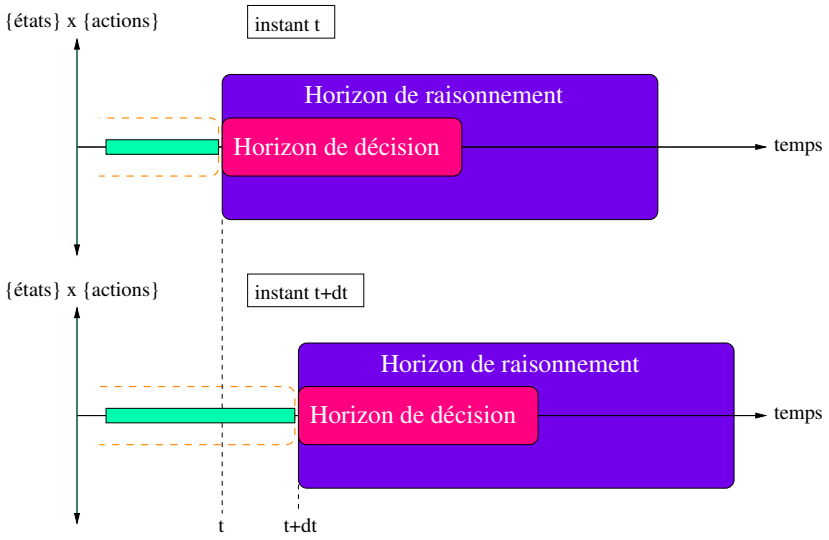


FIG. 1.11: Planification continue

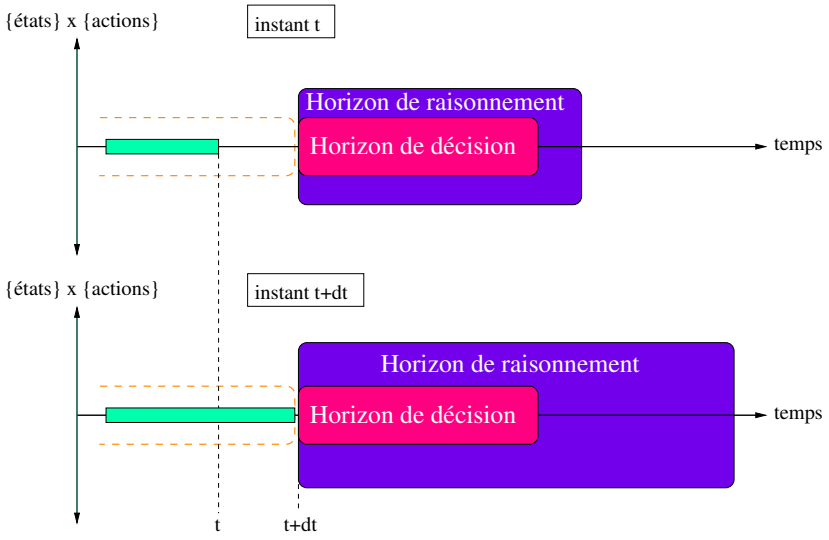


FIG. 1.12: Planification anytime

Enfin, lorsque les instants de décision sont bien définis (c'est le cas dans [Damiani, 2005], où les instants de décisions correspondent aux dates de début des actions d'ob-

servation, ou bien dans notre mission de référence, où le prochain instant de décision correspond à la date de fin de l'action courante), on peut choisir de replanifier à chaque fois qu'une décision doit être prise. Cette approche utilise des algorithmes interruptibles [Zilberstein, 1996] qui adaptent leur durée de raisonnement au temps effectivement disponible pour raisonner. La taille de l'horizon de raisonnement est donc croissante tandis que celle de l'horizon de décision est petite et fixe (voir figure 1.12), permettant au système de bien s'adapter à des contextes très réactifs.

2.4 Sous-problème du problème de planification

2.4.1 Ordonnancement

Concepts généraux L'ordonnancement consiste typiquement à allouer un ensemble de ressources à un ensemble de tâches, en respectant différents types de contraintes matérielles (capacités des ressources) et temporelles (durées des allocations, précédences entre tâches).

Considérons le type de problème suivant, composé de :

- un ensemble $T = \{t_1, ..., t_n\}$ de tâches. Chaque tâche t_i est caractérisée par un temps processeur p_i ;
- un ensemble $R = \{r_1, ..., r_m\}$ de ressources nécessaires pour exécuter les différentes tâches. Ces ressources peuvent être de différents types : renouvelable[4], consommable[5], etc.
- un ensemble $C = \{c_1, ..., c_l\}$ de contraintes. Ces contraintes se divisent en deux types :
  - des contraintes sur les ressources qui limitent leur capacité maximum ;
  - des contraintes temporelles qui contraignent l'exécution des tâches à l'intérieur de fenêtres temporelles.

Résoudre ce problème d'ordonnancement consiste à assigner à chaque tâche une date de début et une date de fin tout en satisfaisant et l'ensemble des contraintes sur les ressources et l'ensemble des contraintes temporelles, et en optimisant éventuellement un certain critère comme le temps total d'exécution.

Intégration de la planification et de l'ordonnancement Dans la littérature, planification et ordonnancement sont traditionnellement étudiés séparément. D'un côté, résoudre un problème de planification consiste à déterminer "comment" atteindre un but donné au travers d'une séquence d'actions et sans considérer les problèmes de durées ou de partage des ressources. De l'autre côté, résoudre un problème d'ordonnancement consiste à déterminer "quand" exécuter un ensemble d'actions de manière à satisfaire les contraintes temporelles et les contraintes sur les ressources.

[4]Une ressource est renouvelable si après avoir été allouée à une ou plusieurs tâches, elle est à nouveau disponible en même quantité.

[5]Une ressource consommable est consommée par une tâche de façon irréversible.

Cependant, planification et ordonnancement sont clairement liés, c'est pourquoi les recherches actuelles tendent à intégrer ces deux concepts. Une solution actuelle consiste à utiliser une représentation commune pour un problème de planification et d'ordonnancement, une représentation sous forme de problème de satisfaction de contraintes (Constraint Satisfaction Problem) [Montanari, 1974]. C'est le cas de systèmes tels que HSTS [Muscettola *et al.*, 1992] et IxTeT [Ghallab et Laruelle, 1994]. Il existe également d'autres systèmes, comme GraphPlan [Blum et Furst, 1995] ou CRIKEY [Halsey *et al.*, 2004], basés essentiellement sur un planificateur classique, enrichi avec des fonctionnalités d'ordonnancement (prise en compte du temps et des ressources).

2.4.2 Planification de mouvement

Les techniques de planification actuelles permettent de résoudre des problèmes académiques toujours plus complexes. Cependant, lorsque l'on s'intéresse à des applications réelles, leurs performances sont souvent revues à la baisse. Que ce soit pour un bras articulé sur une chaîne de production, pour un rover d'exploration martien, ou un satellite agile d'observation de la Terre (voir le chapitre 2), la gestion des mouvements d'un engin mobile complique énormément la tâche des planificateurs.

Même si cet aspect mécanique de la planification peut être noyé dans un algorithme de planification spécialisé, on trouve souvent dans la littérature, une distinction entre le planificateur de tâches, haut niveau, et le planificateur de mouvement plus bas niveau. Les interactions entre ces deux planificateurs sont ensuite réalisées de différentes façons :

- l'approche classique utilisée lors de la conception d'une architecture robotisée [Latombe, 1991] consiste à définir une hiérarchie entre ces planificateurs : un plan d'activités est généré par le planificateur de tâches, et ce plan est ensuite raffiné, d'un point de vue mécanique, par le planificateur de mouvement. Si aucune solution n'est trouvée par le planificateur de mouvement, le planificateur de tâche doit déterminer un nouveau plan d'activités ;
- cette approche peut être améliorée en mémorisant les actions nécessitant les services du planificateur de mouvement, et en permettant au planificateur de tâches de revenir sur le choix de ces actions dans le cas où elles sont irréalisables mécaniquement ;
- une troisième approche consiste à entrelacer complètement les deux planificateurs [Guitton *et al.*, 2008]. Pour construire son plan d'activités, le planificateur d'activités fait régulièrement appel au planificateur de mouvement pour vérifier la faisabilité de certaines actions.

Problème de planification de mouvement Que ce soit pour un robot terrestre mobile, ou pour un satellite agile en orbite autour d'une planète, un problème de planification de mouvement classique consiste à déterminer un mouvement continu reliant une configuration de départ de l'engin à une configuration de fin, en évitant les éventuelles collisions avec des obstacles et en respectant les contraintes cinématiques imposées par les actionneurs de l'engin.

Méthodes de résolution La plupart des techniques de planification de mouvement ont recours au même outil mathématique afin de représenter l'état d'un système : l'espace des configurations. Il s'agit d'un espace vectoriel de dimension finie dont chaque dimension représente un degré de liberté du système. Ainsi, à chaque vecteur de l'espace des configurations correspond un état du système.
Par exemple, un solide en déplacement libre dans l'espace possède six degrés de liberté : trois translations et trois rotations ; son espace des configurations est donc de dimension six.

Il existe deux principales catégories de méthodes pour la planification de mouvement [LaValle, 2006] :

- La première est composée de méthodes déterministes complètes. Les algorithmes les plus connus sont les suivants : champs de potentiel, décomposition cellulaire, diagrammes de Voronoï ;
- La seconde catégorie est composée des méthodes probabilistes qui ne sont pas complètes, mais garantissent de trouver une solution s'il en existe une, lorsque la durée de raisonnement dont elles disposent tend vers l'infini (garantie probabiliste). Ces méthodes ne trouveront pas forcément le même chemin à chaque exécution, même avec des conditions initiales similaires. Les méthodes probabilistes peuvent être subdivisées en deux sous catégories : les méthodes d'échantillonnage et les méthodes de diffusion. Les premières cherchent à approximer l'environnement de l'objet qui doit être déplacé, afin de construire une carte réutilisable (de la même manière qu'une carte routière). Les secondes effectuent une recherche aléatoire dans l'environnement, jusqu'à trouver la configuration finale désirée. Les algorithmes les plus connus sont les suivants : algorithme du fil d'Ariane, probabilistic roadmap (PRM) pour les méthodes d'échantillonnage, rapidly-exploring random trees (RRT) pour les méthodes de diffusion.

3 Modélisation et résolution des problèmes de planification

3.1 Planification classique

3.1.1 Représentation

L'objectif de la planification classique est d'atteindre un ensemble de buts donnés, souvent exprimés sous la forme d'un ensemble de formules logiques entre variables booléennes (ou littéraux). L'état initial du monde est également exprimé sous la forme d'un ensemble de formules logiques, et les différentes actions possibles sont modélisées par ce qu'on appelle des opérateurs STRIPS. La syntaxe d'un opérateur STRIPS est la suivante [Laruelle *et al.*, 1994] : <Nom_de_l'action (Paramètres)><Préconditions><Ajouts><Retraits>, où Préconditions est la liste des littéraux qui doivent être vrais pour que l'action puisse

s'exécuter, Ajouts (respectivement Retraits) est la liste des littéraux positifs (respectivement négatifs) caractérisant les effets que l'action aura sur le monde.

Exemple 1 *Une action d'orientation d'un engin spatial vers une cible lointaine se mettra sous la forme suivante :*

<Orienter (?cible)>
<Pointer(?direction), ?direction ≠ ?cible>
<Pointer(?cible)><Pointer(?direction)>

Les préconditions et les effets des opérateurs STRIPS sont limités à une conjonction de littéraux. Une version étendue du langage STRIPS, connu sous le nom d'ADL [Pednault, 1989], autorise l'usage de quantificateurs, de disjonctions dans les préconditions et d'effets conditionnels. D'autres langages, comme PDDL3.0 [Gerevini et Long, 2006], sont alors venus enrichir la modélisation des problèmes de planification en prenant en compte un modèle explicite du temps [Smith et Weld, 1999] (actions non plus instantanées), les besoins et les consommations de ressources [Koehler, 1998] et un modèle d'incertitude pour les effets des actions.

3.1.2 Techniques de résolution

En planification classique, on distingue principalement deux méthodes de recherche de solution :

- une recherche dans l'espace d'états : méthode la plus ancienne qui consiste à appliquer les opérateurs d'action à partir de l'état initial et à essayer d'atteindre l'état but (recherche en avant), ou bien à partir de l'état but et essayer de revenir à l'état initial (recherche arrière), ou encore d'utiliser les deux sens de recherche simultanément ;
- une recherche dans l'espace des plans partiels : méthode qui consiste à partir d'un plan comportant des défauts (objectifs non atteints, actions incompatibles, actions désordonnées, ...) et à lui appliquer des contraintes ou lui ajouter des opérateurs afin d'obtenir un plan valide.

Les techniques algorithmiques de recherche sont très diverses :

Recherche gloutonne Cette technique consiste à construire une solution par des choix successifs sans jamais revenir sur les choix effectués. Ces choix sont souvent guidés par des connaissances a priori sur le problème à résoudre (heuristiques). C'est une technique très rapide et peu consommatrice en mémoire, mais sous-optimale et dont la distance de la solution à l'optimum est fortement dépendante de l'heuristique utilisée. Il est possible d'améliorer la qualité des solutions en effectuant certains choix aléatoirement et en itérant la recherche [Minton *et al.*, 1994]. Enfin, d'autres techniques bruitent l'heuristique en définissant pour chaque choix la probabilité de suivre ou non l'heuristique [Bresina, 1996].

Recherche arborescente Cette technique consiste à construire une solution en ayant la possibilité de revenir sur les choix effectués. L'ensemble des combinaisons de choix (arbre de recherche) est exploré au moyen de différentes techniques : recherche de type profondeur d'abord, largeur d'abord, meilleur (au sens d'un certain critère) d'abord. Cette technique à l'avantage de fournir une solution optimale, mais sa complexité temporelle est exponentielle dans le pire cas.

Recherche locale Sous le terme de recherche locale [Aarts et Lenstra, 1997] sont regroupées un ensemble de techniques (méthode du gradient, recherche tabou, recuit simulé, ...) consistant à partir d'une solution complète obtenue rapidement et à l'améliorer successivement. Ces techniques n'apportent aucune garantie en terme d'optimalité de la solution, mais sont peu consommatrices en mémoire et peuvent fournir rapidement des solutions de bonne qualité.

Programmation dynamique Cette technique part de problèmes simples à résoudre et les utilise de façon récursive pour résoudre des problèmes plus complexes jusqu'à résoudre le problème complet. Une recherche arborescente avec mémorisation et factorisation des états parcourus peut par exemple être vue comme une méthode de programmation dynamique. Cette technique est donc gourmande en mémoire.

Graphplan Le planificateur Graphplan [Blum et Furst, 1995] recherche un plan solution à partir d'une description de type STRIPS des opérateurs, de l'état initial et du but à atteindre. Cette recherche est réalisée en deux étapes successives :

- une expansion du graphe de planification du problème : à partir de la description des opérateurs et des conditions initiales, chaque opérateur est instancié de toutes les façons possibles ; les actions applicables ainsi obtenues sont utilisées pour construire un nouvel ensemble de littéraux dont le produit cartésien est un ensemble incluant l'ensemble des états atteignables après une étape. Cependant, tous les états ne sont pas compatibles et, même si des actions concurrentes sont permises, tous les états atteignables ne peuvent pas être atteints au même instant : dans le cas d'un rover d'exploration martien, par exemple, une action de déplacement ne peut pas être réalisée en même temps que l'observation d'une roche martienne qui nécessite une position fixe du rover. Graphplan prend en compte cette notion d'incompatibilité en établissant des relation d'exclusion mutuelle (*mutex*) entre les actions incompatibles et les états incompatibles. Puis ce mécanisme est réitéré à partir de ce nouvel ensemble de littéraux jusqu'à la validation d'une condition nécessaire de validité des buts. Les règles d'exclusion mutuelle sont les suivantes :
  - Deux actions sont en exclusion mutuelle à une certaine étape si :
    - elles ont des effets opposés ;
    - un effet de l'une s'oppose à une précondition de l'autre ;
    - elles ont des préconditions en exclusion mutuelle à cette étape.

  - Deux états sont en exclusion mutuelle à une certaine étape si :
    - ils correspondent à deux littéraux opposés ;
    - si deux actions permettant de les atteindre sont en exclusion mutuelle à l'étape précédente.
- l'extraction d'une solution par une recherche arrière dans le graphe de planification : cette recherche s'effectue en tenant compte des actions, des littéraux et surtout des contraintes entre les actions qui ont été détectées à l'étape d'expansion du graphe. Si aucun plan solution n'est trouvé, on continue à développer le graphe de planification.

SAT La planification par satisfiabilité (SAT) utilise le langage STRIPS pour représenter un problème de planification et la logique propositionnelle pour le résoudre. Le problème est initialement traduit en un problème de satisfiabilité, c'est à dire sous la forme d'une CNF[6]. Une fois traduit, le problème de planification consiste à déterminer s'il existe une valuation de l'ensemble des variables propositionnelles telle que la CNF soit logiquement vraie.
Étant donnés une longueur fixée du plan n et un ensemble V de variables d'état booléennes décrivant l'état du système, on crée les variables propositionnelles v^i pour tout $i \in \{0, ..., n\}$ et toute variable $v \in V$. La variable v^i est vraie si la variable d'état correspondante est vraie après l'action numéro i. On crée également les variables a^i pour tout $i \in \{1, ..., n\}$ et toute action a. La variable a^i est vraie si l'action numéro i est a. Il est alors possible de transformer le modèle du système en un ensemble de clauses. Par exemple, si l'action a fait passer la variable v^1 à vrai lorsque celle-ci est fausse, alors la CNF contiendra une clause $(\neg v^0) \Rightarrow a^1 \Rightarrow v^1$ (ce qui est traduit par la clause $v^0 \vee \neg a^1 \vee v^1$). L'affectation trouvée par l'algorithme de satisfiabilité peut être immédiatement traduit en plan.

La planification classique avec des solveurs SAT complets et surtout avec des solveurs SAT incomplets, est très efficace si on connaît la longueur n du plan solution. Si cette valeur n'est pas connue a priori, on peut chercher des plans en incrémentant cette longueur tant qu'un plan solution n'est pas trouvé.

3.2 Planification HTN

La planification HTN (Hierarchical Task Network) est basée sur une représentation hiérarchique des tâches réalisables par un système. Contrairement à la planification classique qui cherche simplement à combiner un certain nombre d'actions pour atteindre un objectif exprimé sous la forme d'une conjonction de littéraux, la planification HTN décompose chaque tâche de haut niveau en un ensemble de tâches primitives de bas niveau afin d'atteindre un objectif souvent spécifié comme une tâche de haut niveau. Par

[6]Une proposition est en Forme Normale Conjonctive (CNF) si elle est composée d'une conjonction de disjonctions.

exemple, le fait d'observer une zone à la surface de la Terre par un satellite d'observation peut être spécifié par une tâche `Observer(?cible,?instrument)`. L'algorithme de résolution développe alors récursivement les tâches de haut niveau en réseaux de tâches de niveau inférieur, en suivant certaines règles de transformation appelées *méthodes*. La figure 1.13 représente une instanciation possible de la tâche `Observer(?cible,?instrument)`. Celle-ci peut être remplacée par le réseau ordonné de trois tâches `Pointer(?cible)`, `Calibrer(?instrument)` et `PrendreImage(?cible,?instrument)`, avec la contrainte supplémentaire que le satellite reste orienté vers la cible jusqu'à la fin de la prise de vue.

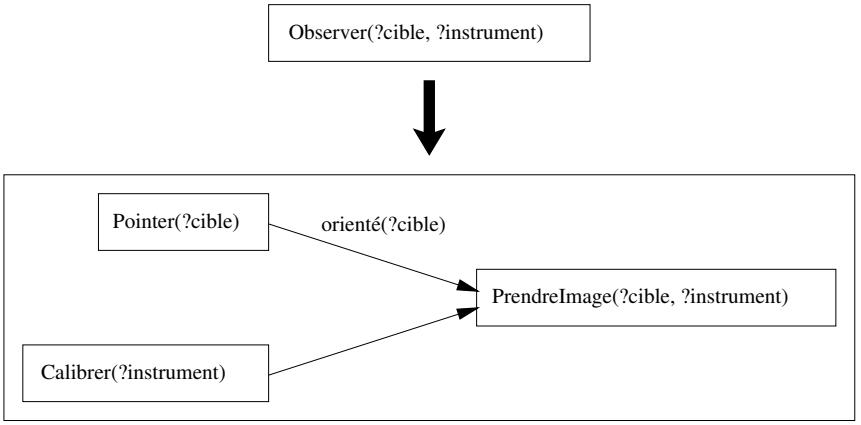


FIG. 1.13: Méthode de décomposition simple de l'action d'observation d'une cible

HTN-Planifier(N)

1. **Si** N contient des conflits
2. **Si** les conflits ne peuvent pas être résolus **alors échec**
   Sinon choisir un moyen de résoudre les conflits et l'appliquer
3. **Si** N ne contient que des tâches primitives, **retourner**(N)
4. Sélectionner une tâche non primitive t dans N
5. Choisir une méthode $t \rightarrow D$ pour la tâche t
6. $N' \leftarrow$ remplacer t par D dans N
7. HTN-Planifier(N')

FIG. 1.14: Procédure de décomposition HTN pour un réseau de tâche N

Après chaque développement, le planificateur HTN détermine les éventuels conflits entre tâches du réseau et tente de les résoudre en imposant des contraintes d'ordre

supplémentaires. La planification se termine lorsque le réseau ne contient plus que des tâches primitives et que l'ensemble des contraintes est cohérent. La figure 1.14 montre le pseudo code d'un algorithme de planification HTN.

3.3 Planification dans l'incertain

La planification traditionnelle permet la génération de plans en adoptant une approche optimiste, c'est-à-dire sans prendre en compte les incertitudes éventuelles dans le raisonnement. Mais pour la conduite de systèmes complexes (dans le cadre de la robotique mobile par exemple) il apparaît nécessaire d'intégrer différentes incertitudes au raisonnement. En effet, l'étude de problèmes de planification réalistes fait intervenir trois types d'incertitudes :

- l'incertitude sur l'état du système : il peut être partiellement observable, c'est-à-dire que la connaissance que l'on en a est limitée par les données des capteurs utilisés ;
- l'incertitude sur l'arrivée d'évènements extérieurs au système et qu'il ne contrôle pas ;
- l'incertitude sur les effets des actions entreprises par le système (non déterminisme).

Face à cela, une première voie a été ouverte dès le milieu des années 80, avec des mécanismes de révision de plans (réparation de plans, réutilisation de plans, ...) [Hammond, 1990; Kambhampati et Hendler, 1989b,a, 1992]. L'enjeu de ces approches est de pouvoir entrelacer efficacement planification d'actions et exécution, en évitant la replanification complète des problèmes.

Une seconde voie de recherche s'est dégagée à partir des années 90 sur la question de la prise en compte explicite dans le processus de planification du caractère incertain de l'environnement lors de l'exécution du plan. Elle a mené à la production de plans "réactifs", "conditionnels" qui permettent de faire dépendre les actions à mener des observations qui seront faites lors de l'exécution, voire aussi de sélectionner les observations à mener en fonction de l'évolution de la situation perçue. Cette approche est bien sûr à relier à la théorie de la commande optimale en automatique, où a été développé le concept de boucle fermée, loi de contrôle liant à chaque instant l'action à appliquer à l'état observé du système, dans le but de définir un mode de conduite du système, robuste aux incertitudes.

3.4 Processus Décisionnels de Markov

Les MDPs (Markov Decision Processes) sont utilisés depuis les années 50 en recherche opérationnelle et théorie de la décision pour modéliser des problèmes de décision séquentielle [Puterman, 1994]. Ils sont aujourd'hui également utilisés par la communauté d'Intelligence Artificielle pour résoudre des problèmes de planification sous incertitudes. Un MDP peut être vu simplement comme un espace d'états dans lequel les transitions entre états sont probabilistes.

Définition 4 Une politique est une fonction de l'ensemble des états à valeur dans l'ensemble des actions qui, à tout état, associe une action à entreprendre.

L'approche classique de résolution des MDPs s'appuie sur la notion de valeur optimale qui représente la valeur espérée en tout état, de suivre une politique optimale à partir de cet état. Deux familles principales d'algorithmes, par itération de la valeur ou itération de la politique, permettent de trouver des politiques optimales qui spécifient, pour chaque état d'un MDP, l'action qui doit être entreprise.

Le principal inconvénient des MDPs en planification est la taille de l'espace d'états. Beaucoup de travaux se sont alors intéressés à la réduction de cet espace d'état soit en utilisant des représentations plus compactes, soit en utilisant des techniques de recherche qui ne vont développer que les parties les plus utiles et les plus probables de l'espace d'états.

Enfin, l'extension du cadre des MDPs à la prise en compte d'observations partielles a été abordée dans le cadre des processus décisionnels de Markov partiellement observables (POMDP). En plus des effets incertains des actions entreprises, les POMDPs disposent d'une distribution de probabilité sur leurs états. Cependant, les méthodes algorithmiques développées jusqu'alors ne sont efficaces que sur des problèmes de très petite taille.

Chapitre 2

Contexte

1 Le projet AGATA

1.1 Présentation

Aujourd'hui la majorité des engins spatiaux sont téléopérés. Des plans d'activités extrêmement précis sont régulièrement téléchargés du sol vers l'engin et la part d'autonomie qui lui est laissée est limitée au strict minimum : régulation thermique, contrôle routinier de l'orbite ou de l'énergie... Cependant, les fenêtres de visibilité et donc de communication d'un satellite défilant, tel qu'un satellite d'observation de la Terre, avec une station sol sont rares et de courte durée. Les temps de communication d'une sonde interplanétaire avec une station de réception terrestre peuvent être très longs[1] et incompatibles avec les temps de réaction demandés de la mission.

Doter les engins spatiaux d'une autonomie décisionnelle embarquée permettrait d'augmenter leur disponibilité à l'aide de diagnostic de panne et de reconfiguration automatique en cas de défaillance, et augmenterait leur réactivité en leur permettant de s'adapter immédiatement aux conditions courantes :

- modification des objectifs (nouvelle zone à explorer dans le cas d'un robot martien) ;
- modification de l'environnement (modification de la couverture nuageuse au dessus d'une zone à imager par un satellite d'observation) ;
- modification de l'état courant de l'engin (défaillance d'un instrument).

Habitués à contrôler entièrement les engins spatiaux, les différents acteurs du domaine spatial sont encore aujourd'hui réticents à l'idée de laisser plus d'autonomie à leurs satellites, rovers ou autres sondes d'exploration. C'est dans ce contexte que s'est mis en place dès le mois de juillet 2004, un programme de recherche (projet AGATA [Charmeau et Bensana, 2008] : "Architecture Générique d'Autonomie, Tests et Application") commun au CNES, au LAAS et à l'ONERA, portant sur l'autonomie des systèmes spatiaux. Il vise,

[1] De l'ordre d'une dizaine de minutes dans le cas de la sonde Mars Express.

au travers du développement d'un démonstrateur sol de satellite autonome, à démontrer la possibilité d'embarquer des mécanismes traditionnellement utilisés au sol.

1.2 Scénarios cibles

Trois missions ont été définies pour servir de scénarios de tests au projet AGATA : une mission satellitaire de surveillance de la Terre, une mission d'observation de la Terre par un satellite agile, et une mission d'exploration planétaire. Ces missions requièrent toutes un niveau d'autonomie important pour être menées à bien.

1.2.1 Mission satellitaire de surveillance de la Terre

Cette mission de référence s'inspire de projets de missions de type *Earth Watching* de l'ESA[2] et s'intéresse à la surveillance de volcans et de feux de forêts à la surface de la Terre (notamment la mission Fuego [Escorial *et al.*, 2001]).

La priorité de la mission est d'informer les utilisateurs de l'apparition d'incendies ou d'éruptions volcaniques et de leur localisation le plus rapidement possible. Ces sites sont ensuite surveillés régulièrement par le système.

La mission est assurée par un unique satellite, référencé par le nom de Frankie, en orbite circulaire inclinée de 60 ° à 800 km d'altitude.

La charge utile du satellite Frankie, résumée sur la figure 2.1, comporte deux instruments dédiés à l'accomplissement de cette mission :

- un hotspotter, instrument à champ large fonctionnant en permanence, dans une bande de fréquences appartenant aux infrarouges, en mode pushbroom[3] et chargé de la détection de points chauds à la surface de la Terre, en avant du satellite ;
- et un imageur, instrument à champ étroit fonctionnant également en mode pushbroom, et chargé de la réalisation des prises de vues. Par défaut, sa ligne de visée passe par le nadir[4] (pas de dépointage avant ou arrière) et un miroir orientable actionné par un moteur pas à pas lui permet d'atteindre tout point de la zone couverte par l'instrument de détection en provoquant un dépointage latéral de la ligne de visée.

Le fait que la tâche de détection soit active en permanence explique pourquoi le satellite Frankie est un satellite non agile, c'est-à-dire s'interdisant tout mouvement en attitude (roulis, tangage ou lacet).

[2] voir http ://earth.esa.int/ew/

[3] Une barrette de détecteurs dont la projection au sol forme une bande perpendiculaire à la trace au sol du satellite, balaie le sol au fur et à mesure de la progression du satellite sur son orbite.

[4] Le nadir est le point de la sphère céleste représentatif de la direction verticale descendante, en un lieu donné (par opposition à zénith).

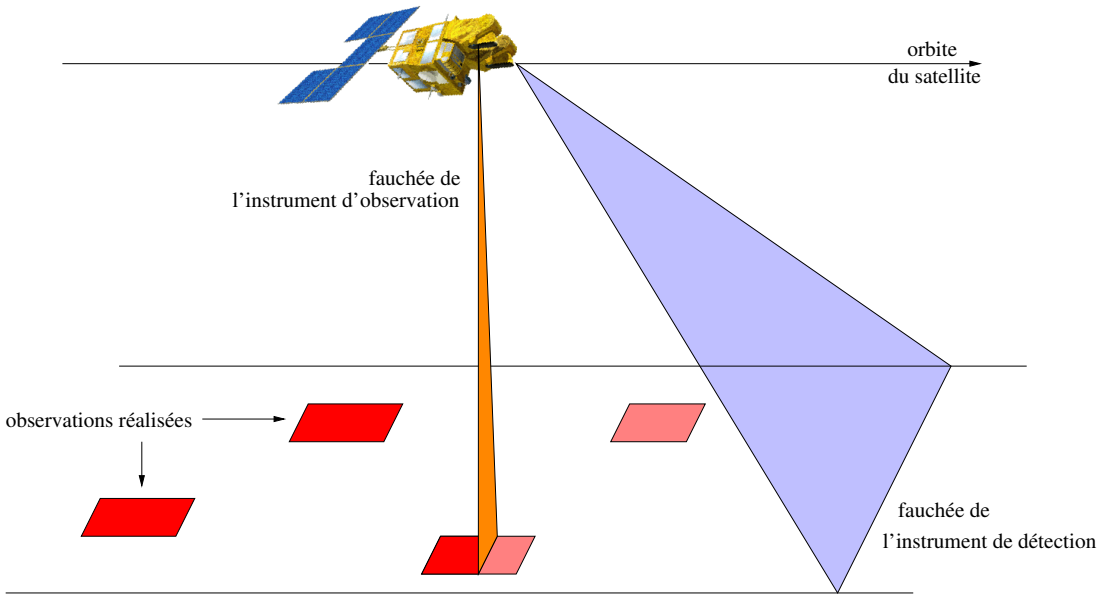


FIG. 2.1: Charge utile du satellite Frankie

1.2.2 Mission d'observation de la Terre par un satellite agile

Cette deuxième mission de référence est une mission satellitaire d'observation de la Terre inspirée de la mission Pléiades - Haute Résolution [Boussarie et Boissin, 2006] du Centre National d'Études Spatiales : un satellite, en orbite autour de la Terre, reçoit régulièrement des requêtes d'observation émises par des organismes demandeurs civils ou militaires. L'objectif du satellite est de satisfaire au mieux les organismes demandeurs en réalisant le plus grand nombre d'observations, et en télédéchargeant les données[5] recueillies aux centres de mission.

Le satellite, placé sur une orbite circulaire inclinée de 98 ° à 700 km d'altitude, est un satellite agile, c'est-à-dire capable de modifier son orientation autour de son centre d'inertie[6]. Cette agilité autorise tous les azimuts d'acquisition d'une prise de vue, y compris dans le sens inverse au défilement du satellite, et augmente ainsi le retour de la mission.

Tous les équipements du satellite, décrits sur la figure 2.2, sont fixes sur la plate-forme :

– pour l'observation d'une zone le satellite utilise un instrument d'observation haute résolution à champ étroit, fonctionnant en mode pushbroom[7] ;

[5]Télédécharger des données consiste à les transmettre du bord (le satellite) vers le sol (les centres de missions).

[6]On appelle *attitude* du satellite son orientation autour de son centre d'inertie.

[7]Cette fois-ci, la projection au sol de la barrette de détecteurs balaie la surface de la Terre selon les mouvements en attitude du satellite.

– un instrument à champ large, de détection de la couverture nuageuse en avant du satellite fonctionne également en mode pushbroom et utilise une matrice de détecteurs à la place d'une simple barrette afin d'accélérer la détection [Lachiver *et al.*, 2001].

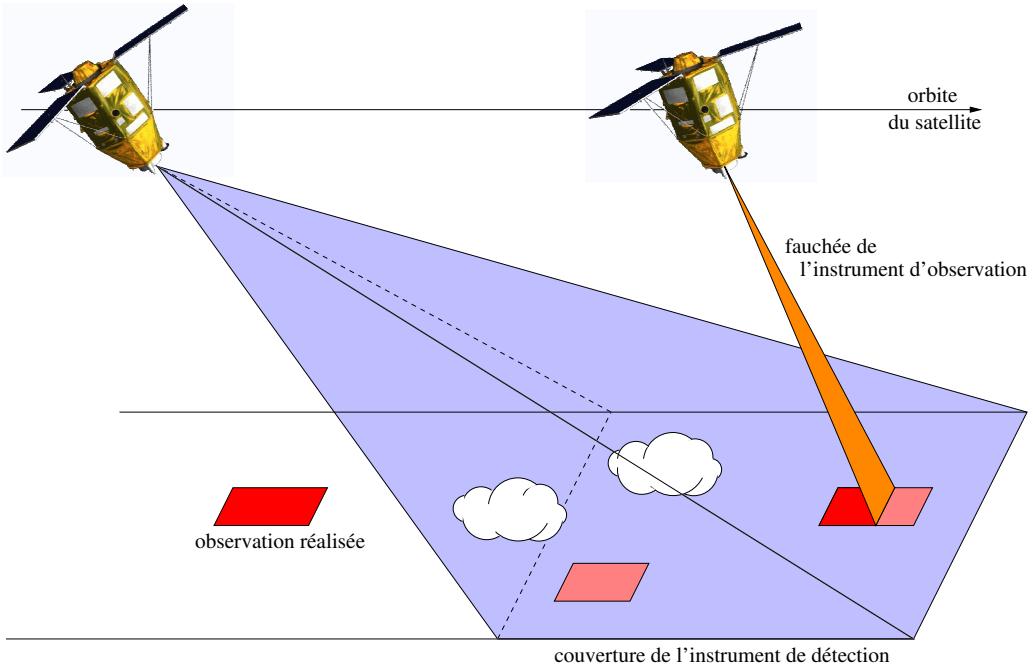


FIG. 2.2: Instruments de détection et d'observation du satellite

1.2.3 Mission d'exploration planétaire

Cette mission s'apparente aux différentes missions d'exploration de la planète Mars du programme Aurora (voir http ://www.esa.int/SPECIALS/Aurora/). La mission met en jeu un engin en orbite autour de la planète (orbiter), et plusieurs engins fixes à la surface de Mars (landers). À l'heure actuelle, ce scénario est le moins avancé du projet AGATA. Le point critique d'une telle mission semble être la gestion des ressources. Cette mission permettra d'étudier notamment les concepts de :

– niveaux d'énergie incertains : par exemple, la poussière qui s'y dépose empêche de connaître précisément l'état des panneaux solaires d'un lander martien ;
– autonomie pour palier à la rareté des fenêtres de communication entre l'orbiter et les landers ou entre l'obiter et les centres de mission terrestres ;
– possibilité d'autoriser l'orbiter à générer de nouvelles requêtes aux landers, sans l'intervention d'un opérateur humain.

Un des principaux objectifs du projet AGATA est de définir une architecture générique commune à toutes ces missions, intégrant niveaux exécutif et décisionnel.

1.3 Architecture générique de contrôle d'un engin autonome

Une des premières réalisations du projet AGATA a était de mettre en place une architecture générique pour le contrôle d'un engin spatial autonome [Verfaillie, 2006].

1.3.1 Qualités attendues de l'architecture

Ce paragraphe dresse une liste des principales qualités requises pour cette architecture.

Généricité L'architecture proposée devait être avant tout une architecture générique. C'est-à-dire qu'elle devait être indépendante du type de mission considéré (mission d'observation, de surveillance, d'exploration, ...), du système à contrôler (système de contrôle de l'orbite du satellite, de l'attitude du satellite, des observations, des télédéchargements, ...), du type de tâche à contrôler (tâche physique ou tâche de traitement de l'information), du type de contrôle à exercer (définition et ordonnancement des tâches à exécuter, décomposition de ces tâches en tâches élémentaires, décisions sur les dates d'exécution, sur les ressources utilisées ou sur des paramètres d'exécution, ...) et de la façon dont ce contrôle est exercé (programme préétabli, exécution d'un plan préétabli avec ou sans possibilité de replanification en ligne, exécution d'une politique préétablie, règles de décision, décision en ligne, ...).

Adéquation au contrôle en boucle fermée L'architecture proposée devait prendre en compte les quatre composants essentiels du contrôle d'un système en boucle fermée :

1. le *suivi de l'état* : suivi de l'état du système et de son environnement ;
2. le *suivi de la consigne* : suivi des objectifs de la mission, incluant les objectifs permanents de survie du système ;
3. le *calcul de la commande* : choix des décisions ;
4. l'*exécution de la commande* : exécution des décisions.

Qualité et réactivité du contrôle Le contrôle d'un système en boucle fermée doit respecter au mieux les deux exigences de :

- *qualité* de la commande : qualité des décisions prises au regard de l'état du système et des objectifs de la mission ;
- *réactivité* de la commande : décisions prises à temps au regard de la dynamique du système et des objectifs.

Modularité Pour des questions de maîtrise de la complexité et de réactivité du contrôle, l'architecture proposée devait permettre autant que possible de décomposer le contrôle du système global en modules de contrôle autonomes et communicants. Avec une telle décomposition, chaque module est responsable d'un sous-ensemble des composantes (variables d'état) du système, aussi bien en termes de suivi de l'état que de contrôle. Par exemple le module de contrôle de l'orbite d'un satellite est responsable du suivi de toutes les composantes de la position orbitale et de leur contrôle via des manœuvres en cas de dérive par rapport à une orbite de référence.

Encapsulation des données et du contrôle Dans le cadre d'une architecture modulaire, une qualité indispensable est l'encapsulation des données et du contrôle au niveau de chaque module. Cela signifie qu'une demande d'information sur l'état d'une composante passe obligatoirement par une requête d'information au module qui en est responsable (ce qui est une garantie de cohérence sur la lisibilité de cette composante à un instant donné, par différents modules de l'architecture). De la même façon, une demande de contrôle sur l'état d'une composante passe obligatoirement par une requête de contrôle au module qui en est responsable, ce qui garantit une gestion des conflits éventuels au niveau de ce module.

Réflexivité De la même façon que les tâches physiques et les tâches de traitement de l'information, les tâches de contrôle peuvent elles aussi prendre du temps, et doivent donc également pouvoir être contrôlées. C'est le rôle de ce qu'on appelle par la suite la *supervision*.

Simplicité L'architecture doit reposer sur des principes et des concepts simples afin d'être aisément comprise par toute personne un tant soit peu familière des domaines de l'informatique et de l'automatique.

Caractère non contraignant L'architecture proposée ne doit pas imposer de contraintes inutiles en termes d'approches, de formalismes, de méthodes ou d'algorithmes pour le suivi de l'état ou la décision. En d'autres termes, l'architecture ne doit pas préjuger de la façon dont le contrôle est concrètement exercé au sein de chaque module.

1.3.2 Architectures existantes

Point de vue de l'architecture Il existe deux types d'architectures suivant qu'elles soient centrées sur la tâche de planification ou celle de supervision.

Architectures centrées sur la planification Ce sont par exemple l'architecture 3T [Bonasso *et al.*, 1997], l'architecture proposée par le groupe RIA du LAAS-CNRS [Alami *et al.*, 1998], ainsi que celle proposée par la NASA pour l'expérimentation du

Remote Agent à bord de la sonde Deep Space One [Muscettola *et al.*, 1998]. Elles sont organisées en trois niveaux :

- un niveau *délibératif* en charge du suivi de la mission, de la génération et du maintien de plans d'actions aptes à satisfaire les objectifs de la mission (couche décisionnelle) ;
- un niveau *réactif* chargé de décomposer les actions de haut niveau, contenues dans les plans du niveau délibératif, en tâches de plus bas niveau et d'en assurer l'activation, le suivi et le compte rendu d'exécution (couche de contrôle d'exécution) ;
- un niveau *fonctionnel* dédié au contrôle temps réel de l'engin, qui assure l'interface entre le niveau délibératif et les capteurs et actionneurs du système physique (couche de contrôle temps réel).

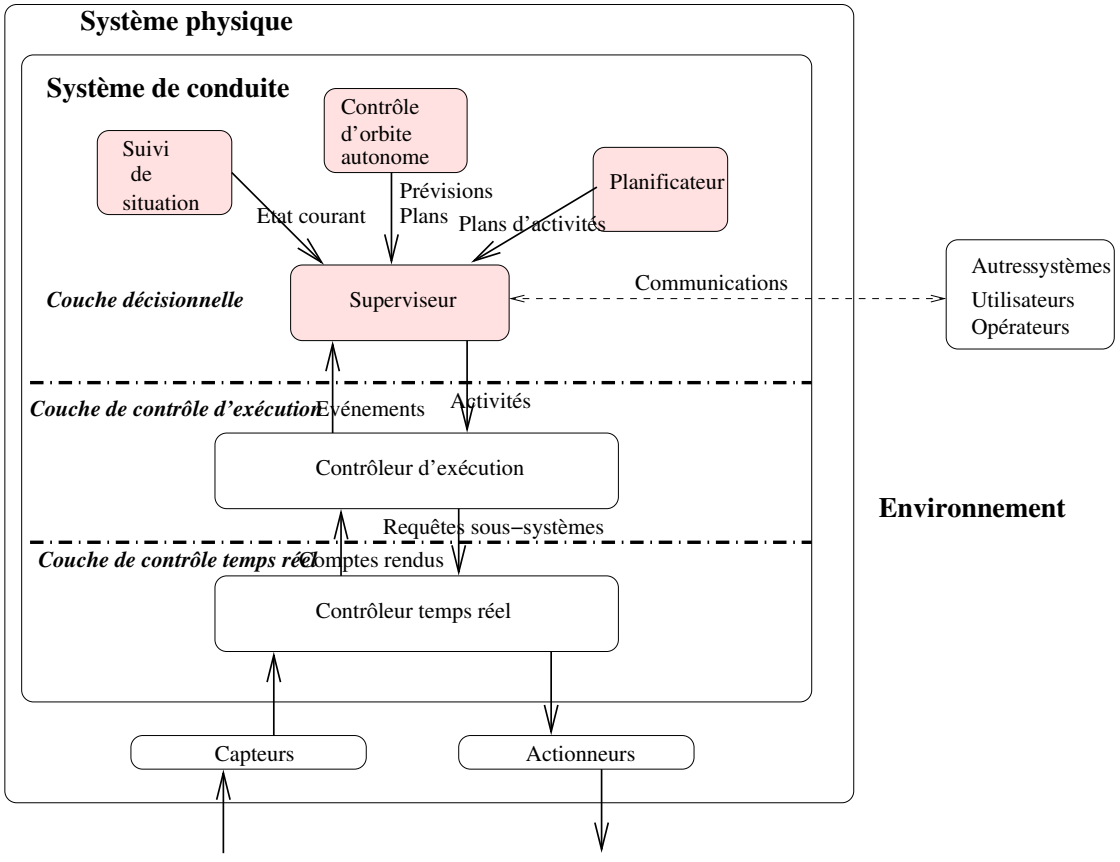


FIG. 2.3: Exemple d'architecture pour la conduite d'un engin autonome, organisée en trois niveaux hiérarchiques

On parle d'architectures centrées sur la planification dans le sens où la tâche de planification est au cœur de l'architecture en trois niveaux : rien ne peut être exécuté au niveau du système physique s'il n'a pas été planifié auparavant, au niveau délibératif.

Architectures centrées sur la supervision Une telle architecture a par exemple été proposée par l'unité de recherche DCSD/CD de l'ONERA [Barrouil et Lemaire, 1999] et utilisée dans le cadre de la gestion de mission d'engins autonomes aériens ou sous-marins. Au cœur de cette architecture, un superviseur préprogrammé est en charge de l'activation et du suivi de toutes les tâches possibles sur le système : aussi bien les tâches physiques (qui sont de niveau fonctionnel dans le type d'architecture précédent) que les tâches de traitement de l'information (qui sont de niveau délibératif dans le type d'architecture précédent).

Modularité de l'architecture En robotique on a observé plusieurs tentatives de développement d'architectures modulaires dans le but d'améliorer leur généricité. C'est par exemple le cas de l'architecture *GENOM* développée par le groupe RIA du LAAS-CNRS [Fleury *et al.*, 1994] qui permet une implémentation modulaire et homogène de l'ensemble de la couche fonctionnelle : chaque module encapsule une fonctionnalité de base du système physique (senseur, actuateur, traitement de données, ...). C'est aussi le cas de l'architecture *IDEA* proposée par NASA-Ames [Muscettola *et al.*, 2002], où chaque module est organisé autour de quatre composants :

- *Model* qui contient les modèles associés aux actions activables par le module ;
- *Plan Database* qui contient toutes les informations sur les actions et les états passés, présents et futurs et qui inclut les buts courants ;
- *Plan Runner* qui peut être vu comme un superviseur du module ;
- *Reactive Planner* qui est en charge d'une planification régulière, réalisée en temps limité sur un horizon temporel limité.

Inconvénients des architectures existantes Quelque soit l'architecture décrite précédemment, le suivi de l'état du système ainsi que le suivi des objectifs courants n'ont pas de place explicitement définie dans l'architecture. De plus, la connexion entre tâches délibératives (suivi de l'état, décision) dont la durée est parfois difficilement maîtrisable, et tâches réactives reste très floue. En particulier dans le cas d'une architecture centrée sur la planification, la complexité de la tâche de planification agit comme un goulot d'étranglement pour le système, nuisant à sa réactivité.

1.3.3 Module générique de contrôle

L'architecture finalement proposée reprend, étend et cherche à améliorer les concepts proposés dans *GENOM* et *IDEA*. La figure 2.4 décrit le schéma d'un module générique de contrôle, lui-même organisé en composants génériques. Le cœur de ce module est un ensemble de quatre composants dédiés au contrôle en boucle fermée : *suivi des requêtes reçues*, *suivi de l'état*, *décision*, *suivi des requêtes émises*.

Le composant *suivi des requêtes reçues* est en charge de la réception et du prétraitement des requêtes émises vers le module, de leur exécution, et de l'émission de comptes rendus de leur exécution. Il maintient un *état courant des requêtes reçues*.

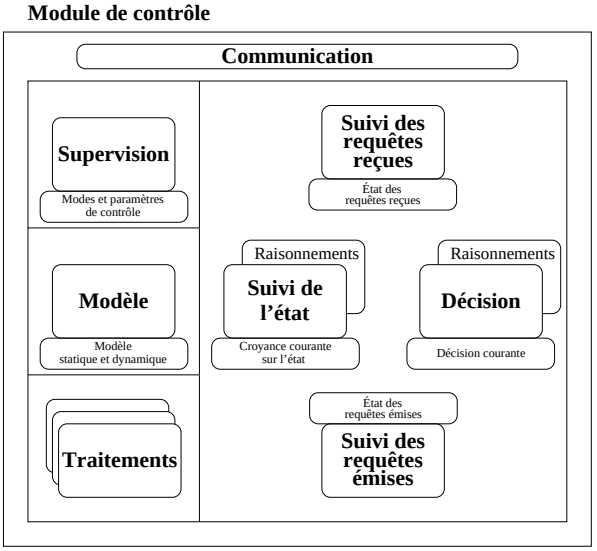


FIG. 2.4: Schéma d'un module générique de contrôle

Le composant *suivi des requêtes émises* est, de la même façon, en charge du prétraitement, de l'émission, du suivi de l'exécution et de la réception des comptes rendus d'exécution des requêtes émises par le module vers d'autres modules, vers des systèmes physiques ou vers des services de traitement de l'information. Il maintient un *état courant des requêtes émises*.

Le composant *suivi de l'état* est en charge du suivi de l'état des composantes du système (variables d'état) dont le module est responsable. Dans le cas le plus général, il s'agit du suivi d'une croyance sur l'état du système, qui utilise des informations en provenance des systèmes physiques et des composants *suivi de l'état* d'autres modules. Il peut faire appel à des raisonnements complexes dont le temps d'exécution n'est pas complètement maîtrisable, mais qui ont a priori un comportement anytime[8]. Il maintient une *croyance courante sur l'état* des composantes dont le module est responsable.

Le composant *décision* est en charge de la décision d'émission de requêtes de contrôle en direction d'autres modules, de systèmes physiques ou de services de traitement de l'information. Pour décider de l'émission de ces requêtes, il utilise des informations en provenance d'autres composants du module (l'état courant des requêtes reçues, l'état courant des requêtes émises, la croyance courante sur l'état), des systèmes physiques et des composants *suivi de l'état* d'autres modules. Comme le composant *suivi de l'état*, il peut faire appel à des raisonnements complexes dont le temps d'exécution n'est pas complètement maîtrisable, mais qui ont a priori un comportement anytime. Il maintient une *décision courante*.

[8]Se dit d'un raisonnement dont le résultat est disponible à tout instant, au moins après un temps minimum d'exécution, et dont la qualité du résultat croit avec le temps.

Au dessus de ces quatre composants de base, un composant *supervision* assure le contrôle du module lui-même (sorte de méta-contrôle) : initialisation, surveillance, et contrôle des différents composants du module.

Le composant *modèle*, optionnel, réunit toutes les caractéristiques statiques (par exemple, la puissance maximale des propulseurs dans le cas d'un module dédié au contrôle d'orbite d'un satellite) et dynamiques (par exemple, les équations liant la position orbitale aux paramètres de propulsion) des composantes de l'état du système dont le module est responsable.

Le composant *traitements*, lui aussi optionnel, réunit tous les services de traitement de l'information liés au module : par exemple, les services de calcul de trajectoire en attitude, pour un module dédié au contrôle d'attitude d'un satellite.

Enfin le composant *communication* assure les échanges de requêtes, d'évènements et de données avec les autres modules, les systèmes physiques ou les services de traitement de l'information.

1.3.4 Exemple de décomposition

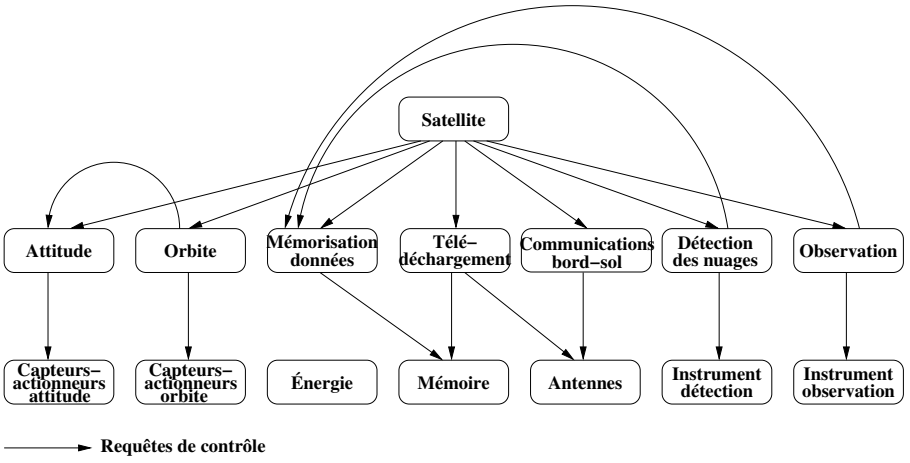


FIG. 2.5: Exemple d'architecture modulaire d'un satellite agile d'observation de la Terre. Requêtes de contrôle inter-modules

La figure 2.5 présente un exemple de décomposition modulaire du contrôle d'un satellite agile d'observation de la Terre, équipé d'un instrument de détection de la couverture nuageuse (cf. la mission décrite dans la partie 2). Les tâches essentielles à réaliser à bord d'un tel satellite, sont les tâches de *détection* de la couverture nuageuse, d'*observation* (observation physique d'une zone au sol, mémorisation et télédéchargement des données), de *communication* avec le sol (réception de requêtes d'observation, émission de données sur l'état du satellite), de *correction d'orbite* (lors d'une déviation du satellite par rapport à son orbite de référence), et de *contrôle en attitude* au cours de chaque action.

Un arc d'un module M vers un module M' traduit une possibilité d'émission de requête de contrôle de M vers M'. Il apparaît dans cette décomposition une hiérarchie naturelle entre modules avec un premier niveau correspondant à la gestion globale du satellite, un second correspondant aux grandes fonctionnalités assurées par le satellite (contrôle de l'attitude et de l'orbite, mémorisation, télédéchargement, communication bord - sol, détection et observation) et un troisième niveau correspondant aux fonctionnalités de base assurées par les senseurs et actuateurs. Des requêtes peuvent être émises d'un module vers un autre de même niveau ou de niveau inférieur. On remarquera qu'aucun arc n'est dirigé vers le module énergie, ce qui correspond au fait que l'énergie n'est pas contrôlée, mais simplement consommée par les différents éléments du système.

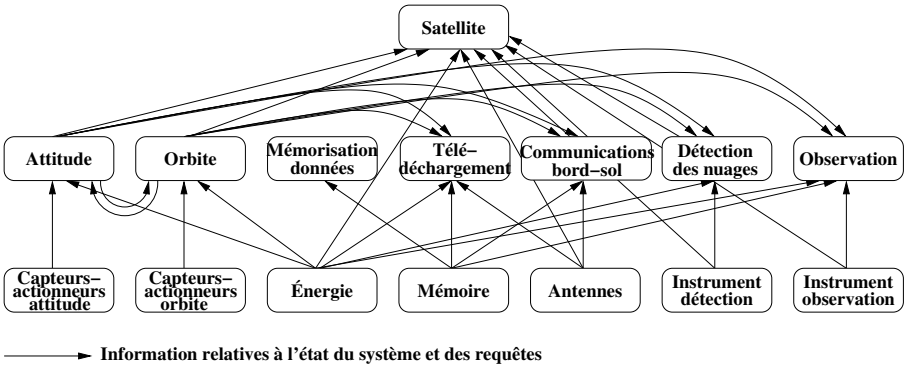


FIG. 2.6: Exemple d'architecture modulaire d'un satellite agile d'observation de la Terre. Échanges de données inter-modules

La figure 2.6 présente la même décomposition avec cette fois les échanges de données entre modules. Un arc d'un module M vers un module M' traduit une possibilité d'échange de données relatives à l'état de M vers M'. On retrouve un mouvement naturel des données du bas vers le haut, en sens inverse des requêtes de contrôle.

Les travaux qui ont été menés dans le cadre de la présente thèse se situent au niveau du module *satellite*, de l'architecture AGATA, responsable de la gestion de la mission dans son ensemble.

1.3.5 Architecture réactive - délibérative développée

Description Dans le cadre d'une mission satellitaire d'observation de la Terre, l'état bord du satellite peut évoluer à tout instant suite à l'arrivée de nouvelles requêtes d'observation ou de nouvelles prévisions météorologiques envoyées par un centre de mission, suite à l'arrivée d'une nouvelle information sur la couverture nuageuse fournie par l'instrument de détection, ou suite à la détection d'un niveau d'énergie ou de mémoire libre inattendu. Dans un environnement si dynamique, l'approche proposée dans [Muscettola

et al., 1998] et consistant à construire un plan d'activités sur un horizon de planification donné, l'exécuter et reconstruire un nouveau plan juste avant la fin de l'horizon de planification, ne semble pas adaptée. Il apparaît plus adéquate de construire un plan d'activités sur un horizon de planification donné et de l'exécuter tant qu'il reste valide, comme cela a déjà été implémenté dans de nombreuses applications [Chien *et al.*, 2000a]. Cependant le principal inconvénient d'une telle approche est qu'il est souvent difficile de vérifier la validité d'un plan dans un environnement dynamique, et particulièrement lorsqu'on cherche à construire un plan optimal. C'est pourquoi nous avons adopté une approche classique dans le domaine de la décision séquentielle : décider, à la fin de chaque action, de la meilleure prochaine action à engager, au vu de l'état connu du système à l'instant de décision.

La manière la plus simple de prendre de telles décisions est d'utiliser des règles de décision prédéfinies. Mais à cause de leur vision locale du problème de décision, même des règles de décision très sophistiquées peuvent produire de mauvaises décisions dans des situations complexes. Une autre manière (utilisée par exemple dans le cadre MDP [Puterman, 1994]) de prendre de telles décisions consiste à calculer hors-ligne une politique optimale (ou proche de la politique optimale) qui associe à chaque état possible du système une action à réaliser. Mais ce genre de méthode est souvent inapplicable dans la pratique, à cause du nombre très élevé d'états atteignables possibles au cours d'applications réalistes comme la notre, et en dépit des nombreuses techniques proposées pour surmonter cette difficulté [Bertsekas et Tsitsiklis, 1996]. Le moyen que nous avons choisi consiste à utiliser des mécanismes de planification, basés sur des algorithmes de planification anytime qui construisent des plans d'activités en partant de l'état de décision et sur un horizon de planification donné, et à sélectionner la première action du meilleur plan trouvé, comme décision à prendre.

L'architecture de contrôle développée dans le cadre du projet AGATA repose sur une distinction claire entre les parties réactive et délibérative du logiciel de vol du satellite, évoluant en parallèle. D'un côté une tâche décisionnelle réactive confère au logiciel de vol de l'engin la capacité de réagir continûment aux stimuli en provenance de l'environnement extérieur, et à la vitesse dictée par celui-ci. Cette tâche est soumise à des contraintes temporelles fortes et ne nécessite que des calculs légers[9]. De l'autre côté, une ou plusieurs tâches décisionnelles délibératives sont en charge du calcul et de l'optimisation des décisions à prendre à bord. Ce sont typiquement des tâches de planification et d'ordonnancement souvent liées à la résolution de problèmes complexes NP-difficiles ou plus, qui nécessitent des calculs lourds et des ressources, en temps de calcul et en mémoire, souvent mal connues à l'avance. Ces tâches délibératives sont contrôlées (activées, terminées, suspendues, reprises ou avortées) à partir des tâches décisionnelles réactives.

Interactions entre tâches réactive et délibérative L'interaction entre tâches réactive et délibérative est basée sur la notion d'*échéance*. Une échéance d'une tâche décisionnelle réactive est un instant futur auquel la tâche doit obligatoirement réagir par

[9]Cet aspect est d'autant plus pertinent que les puissances de calcul embarquées sont faibles.

une prise d'une décision.
Le schéma générique d'interaction [Lemaître, 2007; Lemaître et Verfaillie, 2007] s'appuie sur trois hypothèses :

1. à chaque échéance, il est possible de calculer de manière réactive une décision appropriée, tenant compte
   - de l'état courant de la partie du système à contrôler ;
   - des contraintes de validité et d'intégrité qui s'appliquent à la commande du système ;
   - des délibérations courantes des tâches délibératives associées, si elles existent.
2. une tâche délibérative, au cours de la résolution d'un problème donné, est supposée fournir des délibérations intermédiaires successives de qualités croissantes (comportement anytime) ;
3. à chaque changement significatif, qu'il soit signalé par l'environnemt ou qu'il soit consécutif à une prise de décision, il est possible de calculer la prochaine échéance de manière réactive.

Le fonctionnement de l'interaction réactif - délibératif, représenté sur la figure 2.7, est le suivant :
La tâche réactive :

- reçoit des stimuli de l'environnement, correspondant aux changements d'état pertinents du système ;
- à chaque changement d'état
  - calcule la prochaine échéance ;
  - interrompt et relance les tâches délibératives, en les informant sur les nouvelles données du problème à résoudre.
- à chaque échéance, décide de l'action à engager, sur la base de l'état courant et des dernières délibérations retournées par les tâches délibératives.

Une tâche délibérative :

- déroule une tâche concurrente parallèlement à la tâche réactive qui la contrôle ;
- est avortée et relancée par la tâche réactive à chaque changement d'état ;
- rend des délibérations dès qu'elles sont disponibles et qu'elles améliorent strictement les précédentes.

FIG. 2.7: Schéma des interactions entre l'environnement, une tâche réactive et une tâche délibérative

Contrairement aux architectures de contrôle trois couches largement utilisées en robotique, centrées sur la tâche de planification (rien ne peut être exécuté s'il n'a pas été planifié auparavant), cette architecture réactive - délibérative est centrée sur la supervision (cf. section 1.3), et c'est la tâche réactive qui joue le rôle de superviseur du système.

2 Positionnement de la thèse dans le projet

2.1 Problématique

Dans le cas d'une mission satellitaire d'observation de la Terre de type Spot, deux constatations s'imposent : d'une part, près de 80% des images réalisées par un tel satellite sont rendues inutilisables à cause de la couverture nuageuse qui recouvre les zones observées dans le spectre visible. Pour remédier à cette perte d'efficacité, les ingénieurs considèrent actuellement la possibilité d'équiper ces satellites d'observation avec un instrument de détection de la couverture nuageuse en avant du satellite ce qui permettrait d'affiner le choix des zones à observer. D'autre part, un tel satellite d'observation est rarement en visibilité d'un de ses centres de mission (seulement 10% du temps en moyenne). Afin de tirer parti d'une éventuelle information de détection de la couverture nuageuse à bord du satellite, il devient nécessaire de déplacer les mécanismes de décision (choix des zones à observer notamment) du centre de contrôle au sol vers le satellite.

C'est l'objet de la présente thèse traitant de planification continue pour la conduite d'un satellite d'observation agile autonome [Beaumet, 2006; Beaumet *et al.*, 2006].

2.2 Mission de référence

Dans la section qui suit nous décrivons en détails le scénario de test numéro deux du projet Agata, présenté dans la partie 1.2.2, et qui nous a servi de mission de référence tout au long de la thèse.

Cette mission de référence est une mission satellitaire d'observation de la Terre inspirée de la mission Pléiades - Haute Résolution [Boussarie et Boissin, 2006] du Centre National d'Études Spatiales : un satellite, en orbite autour de la Terre, reçoit régulièrement des requêtes d'observation émises par des organismes demandeurs civils ou militaires. À chacune de ces requêtes sont généralement associées une zone géographique à observer, une priorité et des conditions d'observation (angle de prise de vue, pourcentage maximum de couverture nuageuse admis, contraintes temporelles, ...). Ces observations peuvent être simples - l'observation est réalisée sous un seul angle de prise de vue -, stéréoscopiques (respectivement tri-stéréoscopiques) - l'observation est réalisée deux (respectivement trois) fois avec deux (respectivement trois) angles de prises de vues différents.

2.2.1 Objectifs de la mission

L'objectif de la mission est de satisfaire les organismes demandeurs en réalisant au mieux les observations souhaitées, et en télédéchargeant les données recueillies aux différents centres de mission à la surface de la Terre. Le terme *au mieux* est très vague car plusieurs critères peuvent être optimisés pour une telle mission d'observation : des organismes militaires privilégieront par exemple le nombre important d'observations et la rapidité d'obtention de ces informations, alors que d'autres organismes civils pourront privilégier des critères différents comme la qualité des observations (fonction de l'angle sous lequel les zones sont imagées par le satellite), ou la faible couverture nuageuse présente sur les images.

2.2.2 Structure de la mission

Le satellite agile d'observation

Orbite Le satellite est placé sur une orbite idéale pour une mission d'observation terrestre : c'est une orbite basse (altitude proche de 700 km) permettant une bonne résolution des images réalisées. Elle est également circulaire pour maintenir la distance du satellite à la zone observée quasi-constante, et ainsi limiter les changements de focale de l'instrument d'observation.

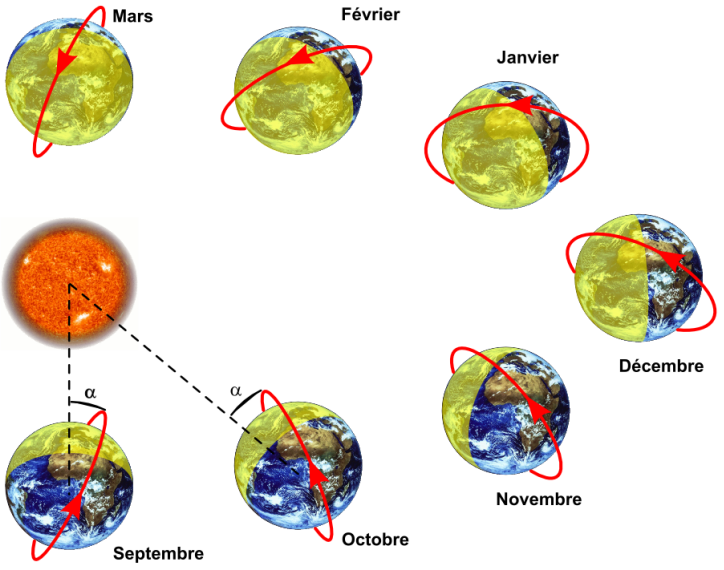


FIG. 2.8: Définition d'une orbite héliosynchrone : l'angle α entre le plan orbital du satellite et la direction Terre - Soleil reste constant au cours du temps

Pour une mission d'observation, et en particulier pour le suivi régulier de l'évolution de certaines zones géographiques (éruptions volcaniques, raz de marée, ...) il est important que le satellite photographie une même zone toujours dans les mêmes conditions d'éclairement ; pour cela il est courant de choisir une orbite héliosynchrone : le satellite passe toujours au dessus d'un même point à la même heure locale (voir figure 2.8). Enfin pour assurer l'héliosynchronisme de l'orbite, il est nécessaire que celle-ci soit quasi-polaire (inclinaison proche de 90 $^\circ$). Combinée à la rotation propre de la Terre, cette orbite permet au satellite de survoler la quasi-totalité de la surface terrestre.

Plate-forme La plate-forme du satellite est organisée autour d'une grappe d'actionneurs gyroscopiques dont le fonctionnement est détaillé dans la section 2.2. Ces actionneurs assurent les mouvements en attitude du satellite en développant des moments cinétiques de l'ordre de 45 N.m.s en roulis et tangage et 20 N.m.s en lacet et des couples de 6 à 7 N.m selon chaque axe.

Lors de la conception d'un tel satellite agile, le choix est souvent fait de garder l'ensemble des équipements fixes sur la plate-forme, afin d'éviter de perturber les mouvements en attitude de l'engin. Le satellite dispose ainsi de trois panneaux solaires plans non orientables fournissant une puissance électrique supérieure à la consommation totale maximale de ses équipements lorsqu'ils sont orientés en direction du Soleil. En période d'éclipse, des batteries, d'une capacité maximale de 5800 Wh, restituent au satellite l'énergie emmagasinée au cours des périodes d'éclairement.

Une antenne non orientable de 64 $^\circ$ d'ouverture, assure les communications entre le satellite et les stations sol au travers d'une liaison haut débit. Une communication entre le satellite et une station au sol n'est possible que lorsque l'intersection entre le cône de visibilité de la station et le cône d'émission de l'antenne n'est pas vide.

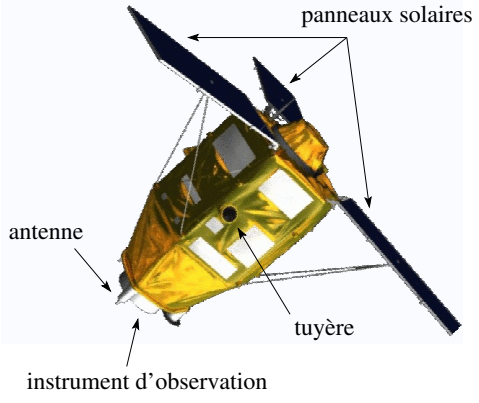


FIG. 2.9: Plate-forme d'un satellite Pléiades

Une mémoire de masse (600 Gbits) permet de stocker l'ensemble des résultats des observations et peut être libérée lors des actions de télédéchargement des données vers les

stations sol, ou lors d'effacement d'images en mémoire. Une petite partie de cette mémoire de masse est également réservée au stockage des informations courantes sur la détection de la couverture nuageuse : ces informations sont écrasées à chaque action de détection.

Enfin trois tuyères, alimentées en ergol, sont disposées à la surface du satellite et assurent le maintien de l'engin sur son orbite tout au long de sa vie.

La figure 2.9 résume l'ensemble des équipements qui constituent la plate-forme d'un satellite Pléiades.

Charge utile La charge utile du satellite est composée de deux instruments optiques représentés sur la figure 2.2. D'une part un instrument d'observation haute résolution à champ étroit de l'ordre de 20 km de fauchée, fonctionnant en mode pushbroom : une barrette de détecteurs balaie la surface de la Terre selon les mouvements en attitude du satellite ; et d'autre part un instrument à champ large, de détection de la couverture nuageuse en avant du satellite, constitué d'une matrice de détecteurs permettant de déterminer quasi-instantanément la couverture nuageuse au dessus d'une zone au sol d'environ 1000 km de côté.

Les centres de mission

Des centres de mission sont répartis sur toute la surface du globe, dans les pays intéressés par le service. Il reçoivent les données de l'instrument d'observation du satellite lorsque celui-ci traverse leur cône de visibilité. On suppose qu'ils sont tous reliés entre eux via un réseau externe.

Le centre de contrôle

Parmi ces centres de mission, il en existe un qui suit et vérifie régulièrement l'état du satellite et corrige éventuellement les anomalies repérées. Il est également chargé de transmettre au satellite les requêtes d'observation formulées par les centres de mission, ainsi que les prévisions climatologiques et météorologiques calculées au sol.

2.2.3 Réalisation de la mission

Pour mener à bien une telle mission, le satellite dispose des actions suivantes : la réalisation d'une observation, le télédéchargement de données vers un centre de mission, le rechargement de ses batteries en pointant ses panneaux solaires en direction du Soleil, la détection en avant, de la couverture nuageuse, la manœuvre orbitale visant à maintenir le satellite sur son orbite de référence, le changement d'attitude et le pointage géocentrique, lorsqu'il n'a rien d'autre à faire.

Observation À tout instant, le satellite a connaissance d'un ensemble d'observations à réaliser. Ces observations peuvent être simples, stéréoscopiques ou tri-stéréoscopiques, et sont caractérisées par un certain nombre de paramètres fixes (comme leur priorité, la

place mémoire qu'elles requièrent ou leurs fenêtres de visibilité) ou variables (comme leur couverture nuageuse connue à bord du satellite). Comme le satellite avance constamment sur son orbite, il dispose d'un temps limité pour réaliser ces observations. Ainsi le nombre d'observations qu'il peut effectivement réaliser au cours d'une révolution est souvent bien inférieur au nombre total d'observations possibles. Pour sélectionner les zones à observer, le satellite s'appuiera sur les différents paramètres de chacune des observations :

- en privilégiant les observations de forte priorité,
- en maximisant la qualité des observations (et donc l'angle sous lequel elles sont réalisées),
- ou en privilégiant les zones les moins couvertes par les nuages.

À la fin de chaque observation, les données sont compressées et mémorisées en attendant d'être télédéchargées vers un centre de mission.

Télédéchargement À chaque instant, le satellite possède en mémoire un ensemble d'observations réalisées. Afin d'être mises à la disposition des utilisateurs, ces données doivent être télédéchargées vers les centres de mission. Lorsque le satellite traverse le cône de visibilité d'une de ces stations, la communication bord - sol devient possible et le satellite doit décider des données qu'il va transmettre en fonction de différents critères et contraintes :

- une image ne peut être télédéchargée que vers un sous-ensemble de l'ensemble des stations sol de la mission ;
- le nombre d'images télédéchargeables est directement lié à la durée de la fenêtre de communication avec la station sol ;
- certaines images sont plus prioritaires que d'autres.

Détection Le satellite maintient constamment à bord une connaissance sur la couverture nuageuse à la surface de la Terre. Celle-ci lui permet d'affiner ses choix d'observation et de maximiser le retour de sa mission. Les actions de détection de la couverture nuageuse permettent de mettre à jour cette connaissance au dessus des zones détectées. La décision de détecter la couverture nuageuse au dessus d'un ensemble de zones au sol, doit être prise en comparant les gains à détecter puis observer ces zones (perte de temps), ou à observer ces zones sans détection préalable (perte d'information sur la couverture nuageuse et donc risque de faire de mauvais choix d'observation).

Manœuvre orbitale Lorsque le satellite dérive de son orbite de référence, le module de Contrôle d'Orbite Autonome du satellite émet des requêtes de manœuvres orbitales dont la réalisation est obligatoire afin de repositionner l'engin correctement sur son orbite. Ces actions, consommatrices en ergol, sont généralement programmées dans des fenêtres temporelles dont la densité en observations est très faible (comme le passage au dessus des pôles).

Rendez-vous en attitude Que ce soit pour débuter une observation, un télédéchargement de données vers un centre de mission, une détection de la couverture nuageuse en avant du satellite, un rechargement des batteries ou un pointage géocentrique, le satellite a besoin de se placer dans une attitude précise. Les actions de rendez-vous en attitude lui permettent de relier ces différentes actions entre elles.

Rechargement des batteries Le rechargement des batteries est réalisé en orientant les panneaux solaires du satellite (et donc le satellite lui-même) en direction du Soleil.

Pointage géocentrique L'action de pointage géocentrique est l'action par défaut du satellite. Elle est exécutée lorsque le satellite n'a rien d'autre à faire (en période d'éclipse, et loin de tout centre de mission par exemple).

Chapitre 3

Gestion de la mission : problème de décision et modélisation

1 Problème de décision

1.1 Description du problème de décision

Le problème auquel nous nous intéressons est un problème d'optimisation du retour global d'une mission d'observation de la Terre par un satellite agile. En entrée de ce problème se trouve un ensemble d'observations à effectuer, qui évolue au cours de la vie du satellite : génération de nouvelles requêtes par le centre de contrôle ou abandon de certaines autres.

L'objectif est de mettre en place des mécanismes de décision visant à maximiser le retour de la mission tout au long de la vie du satellite, en prenant en compte les contraintes sur les ressources à bord comme l'énergie et la mémoire disponibles, ainsi que l'évolution de la couverture nuageuse dans l'atmosphère. Pour cela, le satellite dispose des actions suivantes :

- la réalisation d'une observation ;
- le télédéchargement de données vers une station sol ;
- le rechargement de ses batteries ;
- la détection, en avant, de la couverture nuageuse ;
- le pointage géocentrique ;
- la réalisation d'une manœuvre orbitale visant à corriger son orbite ;
- la réalisation d'un rendez-vous en attitude.

1.2 Nature de la décision

Une manière de maximiser le retour de la mission est de déterminer à chaque instant la meilleure prochaine action que le satellite doit engager.

Espace de recherche Un état du système est défini non seulement par l'état du satellite en termes de niveaux de ressources, d'attitude et de position orbitale (une dizaine de variables d'état), mais également par l'état de son environnement (notamment l'état de la couverture nuageuse) et celui de ses objectifs : ensembles des observations non encore réalisées, réalisées et télédéchargées. Cette multiplicité des variables nécessaires à la description d'un état du système implique une taille conséquente de l'espace des états atteignables qu'il est impossible de parcourir entièrement en un temps raisonnable.

Exemple 2 *Dans le cas du satellite considéré dans notre étude, et dont les caractéristiques sont données au chapitre 6, on peut discrétiser les domaines des différentes variables d'état de la façon suivante :*
Le temps *évolue au cours de l'horizon de raisonnement, entre une date de début (*0 *seconde) et une date de fin (*86400 *secondes par exemple), avec un pas de discrétisation de* 1 *seconde.*
État du satellite *:*

- *le niveau de mémoire libre évolue entre* 0 *et* $6 \cdot 10^{11}$ *bits, avec un pas de discrétisation de* 10^6 *bits ;*
- *le niveau d'énergie disponible évolue entre* 0 *et* 5800 *Wh, avec un pas de discrétisation de* 1 *Wh ;*
- *le niveau d'ergol disponible évolue entre* 0 *et* 80 *litres, avec un pas de discrétisation de* 0.1 *litre ;*
- *l'attitude du satellite peut être définie par trois angles qui évoluent chacun entre* 0 *et* 360 *degrés, avec un pas de discrétisation de 1 degré ;*
- *les vitesses de rotation du satellite en roulis et tangage évoluent entre* $-\frac{45}{850}$ *et* $\frac{45}{850}$ *rd.s*$^{-1}$*, avec un pas de discrétisation de* $\frac{1}{850}$ *rd.s*$^{-1}$ *;*
- *la vitesse de rotation du satellite en lacet évolue entre* $-\frac{20}{750}$ *et* $\frac{20}{750}$ *rd.s*$^{-1}$*, avec un pas de discrétisation de* $\frac{1}{750}$ *rd.s*$^{-1}$*.*

État des objectifs *:*
Pour chaque observation élémentaire,

- *la valeur de la couverture nuageuse au dessus de la zone évolue entre* 0 *et* 1 *avec un pas de discrétisation de* 0.1 *;*
- *la date de début de réalisation évolue dans une fenêtre temporelle d'une centaine de secondes en moyenne, avec un pas de discrétisation de* 1 *seconde ;*
- *la date de début de télédéchargement évolue dans une fenêtre temporelle de* 500 *secondes en moyenne, avec un pas de discrétisation de* 1 *seconde ;*

Si on suppose qu'il y a 400 requêtes d'observation au cours du scénario, le nombre n, d'états atteignables ainsi discrétisés sur l'horizon de raisonnement est alors de :

$$\begin{aligned} n &= \frac{6 \cdot 10^{11}}{10^6} \times \frac{5800}{1} \times \frac{80}{0.1} \times 3\frac{360}{1} \times 2\frac{\frac{45}{850} + \frac{45}{850}}{\frac{1}{850}} \times \frac{\frac{20}{750} + \frac{20}{750}}{\frac{1}{750}} \times 400 \times \frac{1}{0.1} \times \frac{100}{1} \times \frac{500}{1} \\ &= 6 \cdot 10^5 \times 5800 \times 800 \times 1080 \times 180 \times 40 \times 400 \times 10 \times 100 \times 500 \\ n &\simeq 4.3 \cdot 10^{27} \end{aligned}$$

Incertitudes Bien que l'environnement spatial impose un comportement plutôt déterministe des actions du satellite (pas de perturbation due au frottement de l'air, ou au passage de nuages entre le Soleil et les panneaux solaires du satellite par exemple), il subsiste encore certaines incertitudes dans le déroulement de la mission : les statistiques climatologiques et les prévisions météorologiques actuelles ne permettent pas de prédire suffisamment précisément les conditions d'ennuagement de chaque point à la surface de la Terre, et les résultats d'une action de détection de la couverture nuageuse en avant du satellite sont soumis à une très forte incertitude. De plus, la place mémoire nécessaire pour mémoriser une image est difficilement prévisible car elle dépend du taux de compression appliqué à l'image, ce taux dépendant lui-même des caractéristiques de l'image. Enfin, l'arrivée d'une nouvelle requête suite à une nouvelle demande d'un utilisateur au sol, ou bien la génération par le système de contrôle d'attitude et d'orbite du satellite, d'une requête de manœuvre orbitale suite à la dérive du satellite par rapport à son orbite de référence, sont difficilement prévisibles.

La taille de l'espace de recherche et les incertitudes, parfois grandes, sur l'évolution du système nous orientent vers :

- un mécanisme de prise de décision prenant en compte toutes les informations disponibles sur l'état courant de l'engin (position orbitale, attitude, niveaux de ressources, ...), les observations envisageables à partir de l'état courant, les prévisions météorologiques et les informations issues de la détection de la couverture nuageuse, afin de limiter la taille de l'espace des états atteignables ;
- un raisonnement sur un horizon limité pour, en plus de limiter l'espace des états atteignables, palier à l'absence de modèle d'incertitude sur l'arrivée de nouvelles requêtes d'observation ou de manœuvres orbitales ;
- un oubli volontaire de ces incertitudes dans le raisonnement et une remise en cause de la décision à chaque fois que la fin de l'horizon est atteinte ou à chaque modification de l'état courant.

1.3 Contraintes sur les mécanismes de décision

Décision globale Le satellite considéré est un satellite agile. Son attitude peut être vue comme une ressource partagée par l'ensemble de ses activités : pour s'exécuter chaque action nécessite une trajectoire en attitude précise du satellite, empêchant ainsi toute autre action en parallèle, en dehors des actions de télédéchargement opportunistes lorsque le satellite est en visibilité d'une station sol de réception (voir figure 3.1). La faisabilité d'une action à un instant donné sera conditionnée en particulier par l'attitude de fin de l'action précédente. C'est pourquoi il n'est pas envisageable, dans les mécanismes de décision, de raisonner séparément sur chaque type d'activités comme cela a été fait dans [Damiani, 2005]. Il est nécessaire de prendre des *décisions globales* en raisonnant sur l'ensemble des activités du satellite.

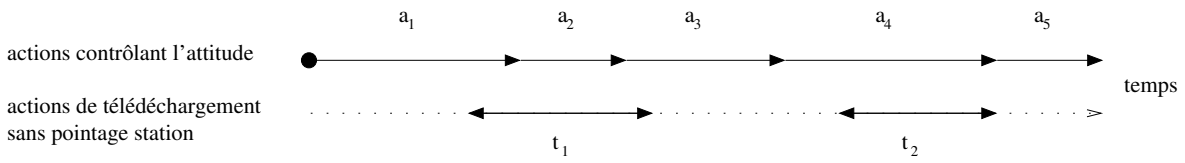


FIG. 3.1: Comportement haut niveau du satellite

Décision en ligne La diversité des situations rencontrées en terme de répartition des requêtes d'observation et d'évolution de la couverture nuageuse, empêche d'emblée toute construction hors ligne d'une politique envisageant tous les états possibles par lesquels pourrait passer le satellite.

Afin de tirer avantage de l'instrument de détection de la couverture nuageuse en avant du satellite, les mécanismes de décision doivent être appliqués *en ligne*, juste au moment où l'information sur la couverture nuageuse devient disponible.

Décision à bord Par définition, un satellite défilant d'observation de la Terre n'est pas en visibilité permanente d'un centre de contrôle à la surface de la Terre (voir figure 3.2). Les décisions ne peuvent donc plus être prises au sol, puis transmises au satellite, mais elles doivent être prises *à bord* du satellite, afin de prendre en compte les informations disponibles sur la couverture nuageuse fraîchement détectée.

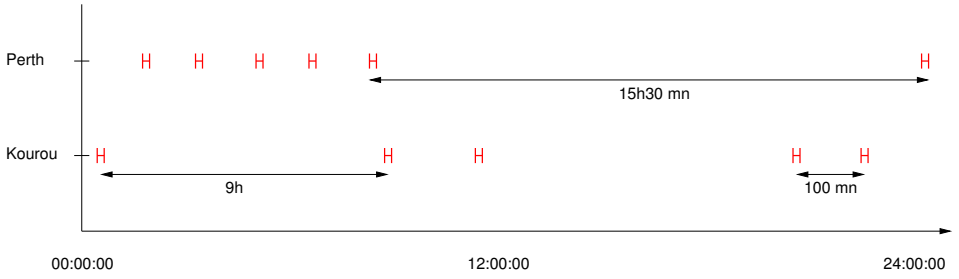


FIG. 3.2: Exemple de visibilités de deux stations sol sur 24h, pour un satellite d'observation de type Spot

Décision en temps contraint L'agilité du satellite lui permet d'une part de réaliser des observations en pointant son instrument d'observation, et donc le satellite lui-même, 47 ° en avant, et d'autre part de détecter la couverture nuageuse en pointant son instrument de détection 30 ° en avant. Par conséquent, la fenêtre de réalisation d'une observation peut débuter immédiatement après l'arrivée d'une information sur la couverture nuageuse des observations à venir. La décision de réaliser ou non une telle observation doit donc pouvoir être prise très rapidement. C'est pourquoi les mécanismes de décision à bord du satellite doivent être capables de fournir rapidement des solutions de bonne qualité.

1.4 Choix du mécanisme de décision

Les différentes approches que nous avons considérées sont une recherche arborescente, une programmation dynamique, une recherche locale et une recherche gloutonne. Une recherche arborescente a vite semblé irréaliste au vu de la taille de l'arbre de recherche : pour chaque action décidée, il est nécessaire de fixer un certain nombre de paramètres comme sa date de début, la durée d'un rechargement des batteries, les données d'observation à télédécharger, l'attitude visée par un rendez-vous en attitude, ... Même si une programmation dynamique semble moins coûteuse en terme de taille de l'espace de recherche parcouru (grâce au mécanisme de mémorisation), elle nécessite en fait une discrétisation de l'état du satellite (non seulement ses niveaux de mémoire, énergie, ergol, son attitude et sa position orbitale, mais aussi la description de ses objectifs courants, c'est-à-dire ses observations télédéchargées, réalisées et restantes) afin de mémoriser sa valeur associée et de la comparer à celles des autres états, et fait exploser la taille de l'espace mémoire nécessaire. Un plan d'activités du satellite est une séquence d'actions au cours de laquelle le satellite suit une trajectoire en attitude très précise (pointage du Soleil dans le cas d'un rechargement des batteries, ou trajectoire plus complexe au cours de la réalisation d'une observation). Chaque action débute dans l'attitude de fin de l'action précédente. À partir de cette constatation, il apparaît difficile, voire impossible, de supprimer, ajouter, ou déplacer une action du plan sans devoir recalculer toute la trajectoire en attitude nécessaire pour toutes les actions suivantes, en supposant qu'elles restent toutes encore possibles. Cette contrainte empêche donc toute utilisation de mécanismes de recherche locale, et privilégie des mécanismes de construction de plans à partir de l'état de décision et en avançant dans le temps. Le mécanisme de décision que nous avons finalement retenu est basé sur un algorithme glouton stochastique itéré [Cicirello et Smith, 2005], qui construit des plans d'actions de manière gloutonne, prenant aléatoirement des décisions en suivant un certain biais heuristique. La solution rendue par un tel mécanisme sera la première décision du meilleur plan construit. Ce mécanisme est décrit plus en détails dans la section 3 du chapitre 5.

2 Modélisation du problème de gestion d'un satellite agile d'observation de la Terre

Dans cette section nous modélisons le problème de gestion d'un satellite agile d'observation de la Terre dans le cadre CNT (Constraint Network on Timelines [Verfaillie *et al.*, 2009]), qui est un cadre générique de modélisation de systèmes dynamiques à évènements discrets à base de variables et de contraintes, telles qu'elles sont définies dans le cadre CSP (Constraint Satisfaction Problem [Rossi *et al.*, 2006]). Les données du problème sont les suivantes :

1. un *horizon de planification* défini par une date de *début* STA et une date de *fin* END,
2. la prochaine date de fin d'éclipse après la date END, END_{ec} ($END_{ec} = END$ si le satellite est en éclipse à la date END),
3. des *caractéristiques du satellite* telles que :
   - la *période nodale* de l'orbite TN,
   - le niveau de *mémoire* maximal disponible MM_{max},
   - le niveau d'*ergol* maximal disponible ER_{max},
   - les niveaux d'*énergie* minimum et maximum EN_{min} et EN_{max},
   - les *puissances* consommées par la plate-forme, par l'instrument d'observation, de détection et l'antenne de télédéchargement P_{sat}, P_{ob}, P_{detec}, P_{dl},
   - la *durée* d'une action de détection de la couverture nuageuse DD.
4. un *état initial* du satellite défini par :
   - des niveaux de *mémoire*, d'*ergol* et d'*énergie* disponibles MM_0, ER_0 et EN_0,
   - une *attitude* du satellite ATT_0, définie par sept composantes : quatre pour le quaternion définissant l'orientation du satellite autour de son centre de masse, et trois composantes définissant sa vitesse de rotation propre.
5. un *ensemble d'observations complexes* OC à réaliser et à télédécharger, avec pour chaque observation complexe $c \in OC$
   - une *date d'émission* $EMIT_c$,
   - une *priorité* $PRIO_c$,
   - une *date limite de télédéchargement* $DEAD_c$,
   - des *contraintes de réalisation* $CONS_c$,
   - un *ensemble de stations sol* SS_c vers lesquelles l'observation complexe peut être télédéchargée,
   - un *ensemble d'observations élémentaires* OE_c, avec pour chaque observation élémentaire $e \in OE_c$
     - une estimation de la *couverture nuageuse* CN_e,
     - une date de dernière information sur sa couverture nuageuse $LDETEC_e$,
     - une quantité de *mémoire* requise SZ_e,
     - une *durée de réalisation* DO_e,
     - une *durée de télédéchargement* DDO_e,
     - un nombre de *fenêtres de réalisation* possibles sur l'horizon de planification NO_e, avec pour chaque fenêtre $k \in [1..NO_e]$, sa date de *début* $SO_{e,k}$ et sa *durée* $DO_{e,k}$,
     - des *contraintes de qualité* $CONS_e$,
     - une *observation complexe* associée OBC_e.

   On définit l'ensemble des observations élémentaires $OE = \{e \in OE_c | c \in OC\}$.
6. un *ensemble d'observations élémentaires* déjà réalisées $OER \subset OE$, avec, en plus, pour chaque observation élémentaire $e \in OER$

  - une fenêtre de réalisation NO_e au cours de laquelle e a été réalisée,
  - une date de début de réalisation SO_e,
  - un angle d'observation A_e.
7. un *ensemble d'observations élémentaires* déjà télédéchargées $OET \subset OER$, avec, en plus, pour chaque observation élémentaire $e \in OET$
  - une fenêtre de visibilité NV_e au cours de laquelle e a été télédéchargée,
  - une fenêtre de télédéchargement ND_e au cours de laquelle e a été télédéchargée,
  - une date de fin de télédéchargement ED_e.
8. un *ensemble de manœuvres orbitales* MNV à réaliser, avec pour chaque manœuvre orbitale $m \in MNV$
  - une *date d'émission* $EMIT_m$,
  - une *date de début* SM_m,
  - une *durée* DM_m,
  - une *attitude* de début $SATT_m$,
  - une *attitude* de fin $EATT_m$,
  - une *consommation d'ergol* CER_m.
9. un nombre de *fenêtres d'éclipse* du satellite sur l'horizon de planification NE, avec pour chaque fenêtre $k \in [1..NE]$, sa date de *début* SE_k et sa *durée* DE_k,
10. un *ensemble de stations sol* SS vers lesquelles s'effectuent les télédéchargements de données,
11. un nombre de *fenêtres de visibilité* des stations sol sur l'horizon de planification NV, avec pour chaque fenêtre $k \in [1..NV]$, sa date de *début* SV_k, sa *durée* DV_k et la station de réception au sol associée $STATION_k$.

Variables classiques Pour modéliser ce problème, on utilise d'abord des variables CSP classiques qui peuvent être vues comme des chronogrammes à un seul attribut et une seule étape :

1. pour chaque fenêtre de visibilité $k \in [1..NV]$ et pour chaque fenêtre de télédéchargement $l \in [1..NW_{max}]$ dans k, ses dates de début et de fin $sd_{k,l}$ et $ed_{k,l} \in [SV_k..SV_k + DV_k]$,
2. pour chaque observation élémentaire $e \in OE$, sa fenêtre de réalisation $no_e \in [0..NO_e]$ et ses dates de début et de fin de réalisation so_e et $eo_e \in [SO_{e,no_e}, SO_{e,no_e} + DO_{e,no_e}]$ (la valeur 0 représente l'absence de réalisation),
3. pour chaque observation élémentaire $e \in OE$, ses fenêtres de visibilité $nv_e \in [0..NV]$ et de télédéchargement $nd_e \in [0..NW_{max}]$ (les valeurs 0 représentent l'absence de télédéchargement).

Fonctions utiles Pour rendre l'écriture du modèle plus compacte, nous utiliserons par la suite les fonctions suivantes :

- $trajectoire(t_{deb}, t_{fin}, att_{deb}, att_{fin}, a)$: renvoie la trajectoire en attitude du satellite suivie entre les attitudes att_{deb} à la date t_{deb} et att_{fin} à la date t_{fin} en exécutant l'action a.
- $puissance(traj, t_{deb})$: renvoie la puissance moyenne produite par les panneaux solaires du satellite, si celui-ci suit la trajectoire $traj$ à partir de la date t_{deb}.
- $startObservationAttitude(o, t_{deb})$: renvoie l'attitude nécessaire pour débuter l'observation élémentaire o à la date t_{deb}.
- $endObservationAttitude(o, t_{fin})$: renvoie l'attitude de fin de l'observation élémentaire o à la date t_{fin}.
- $downloadAttitude(s, t)$: renvoie l'attitude nécessaire pour réaliser un télédéchargement vers la station s à la date t.
- $detectionAttitude(t)$: renvoie l'attitude nécessaire pour réaliser une détection de la couverture nuageuse à la date t.
- $sunPointingAttitude(t)$: renvoie l'attitude nécessaire pour réaliser un rechargement des batteries à la date t.
- $geoPointingAttitude(t)$: renvoie l'attitude nécessaire pour réaliser un pointage géocentrique à la date t.
- $respect(CONS)$: renvoie vrai lorsque les contraintes $CONS$ sont respectées, faux sinon.
- $minDur(att_{deb}, att_{fin})$: renvoie la durée minimale de basculement du satellite entre les attitudes att_{deb} et att_{fin}.
- $detectedObs(t_{deb})$; renvoie l'ensemble des observations élémentaires dont la couverture nuageuse peut être détectée au cours d'une action de détection débutant à la date t_{deb}.
- $observationAngle(o, t_{deb})$: renvoie l'angle sous lequel a été réalisée l'observation o ayant débuté à la date t_{deb}.

Timeline Puis on utilise une timeline tl pour représenter l'évolution de l'état du satellite et de ses objectifs au cours du temps, avec :

- trois attributs mm, en, er représentant respectivement les niveaux de mémoire, d'énergie et d'ergol disponibles,
- un attribut att représentant l'attitude du satellite,
- pour chaque observation élémentaire e, un attribut $ldetec_e$ représentant la date de dernière information sur sa couverture nuageuse avant observation,
- un attribut ti représentant l'instant courant.

et les paramètres des actions engagées par le satellite, avec :

- un attribut ac représentant l'action courante du satellite, parmi les actions d'observation, de télédéchargement, de détection, de manœuvre orbitale, de rechargement des batteries, de pointage géocentrique ou de rendez-vous en attitude,

- un attribut *obsc* représentant l'observation courante dans le cas d'une action d'observation,
- un attribut *st* représentant la station sol vers laquelle s'effectue le télédéchargement, dans le cas d'une action de télédéchargement,
- un attribut d représentant la durée totale de l'action courante, dans le cas d'une action de rechargement des batteries, d'une action de pointage géocentrique ou d'un rendez-vous en attitude,
- un attribut $targ\_att$ représentant l'attitude visée dans le cas d'une action de rendez-vous en attitude,
- un attribut m représentant la manœuvre orbitale courante dans le cas d'une action de manœuvre orbitale.

En résumé, la timeline est composée d'au moins deux étapes (correspondant au début et à la fin de l'horizon de planification) :

$$do(ns(tl)) = [2..+\infty]$$

Elle est constituée des attributs suivants :

$$ats(tl) = \{ac, obsc, st, d, targ\_att, m, mm, en, er, att, ti\} \cup \{ldetec_e | e \in OE\}$$

dont les types sont :

$$\begin{aligned} ty(ac) &= ty(obsc) = ty(st) = ty(d) = ty(targ\_att) = state \\ ty(m) &= ty(mm) = ty(en) = ty(er) = ty(att) = ty(ldetec_e) = state \\ ty(ti) &= time \end{aligned}$$

et les domaines respectifs sont :

$$\begin{aligned} do(ac) &= \{OB, DL, DETEC, MAN, SUN, GEO, RDV\} \\ do(obsc) &= OE \cup \{\perp\} \\ do(st) &= SS \cup \{\perp\} \\ do(d) &= \mathbb{R}^+ \cup \{\perp\} \\ do(targ\_att) &= \mathbb{H} \times \mathbb{R}^3 \cup \{\perp\} \text{ (où } \mathbb{H} \text{ est l'algèbre des quaternions)} \\ do(m) &= MNV \cup \{\perp\} \\ do(mm) &= [0..MM_{max}] \\ do(en) &= [EN_{min}..EN_{max}] \\ do(er) &= [0..ER_{max}] \\ do(att) &= \mathbb{H} \times \mathbb{R}^3 \\ do(ldetec_e) &= [0..END] \\ do(ti) &= [STA..END_{ec}] \end{aligned}$$

Contraintes

Dates de début et de fin d'une observation élémentaire Les dates de début et de fin doivent faire partie d'une fenêtre de réalisation ; la date de fin est la date de début augmentée de la durée de l'observation ; en cas de non réalisation, dates de début et de fin sont arbitrairement positionnées à STA :

$$\begin{array}{rll} \forall e \in OE, & (no_e \neq 0) \rightarrow & (SO_{e,no_e} \leq so_e \leq SO_{e,no_e} + DO_{e,no_e}) \\ & & \wedge (SO_{e,no_e} \leq eo_e \leq SO_{e,no_e} + DO_{e,no_e}) \\ & & \wedge (eo_e = so_e + DO_e) \\ & (no_e = 0) \rightarrow & (so_e = eo_e = STA) \text{ (cas } e \text{ non réalisée)} \end{array}$$

Contraintes entre fenêtre de visibilité et fenêtre de télédéchargement En cas de télédéchargement ces fenêtres sont toutes les deux non nulles. Dans le cas contraire, elles sont toutes les deux nulles et $sd_{k,l}$ et $ed_{k,l}$ sont arbitrairement positionnées à la fin de la fenêtre de visibilité :

$$\begin{array}{ll} \forall e \in OE, & (nd_e = 0) \leftrightarrow (nv_e = 0) \\ & (nd_e \neq 0) \rightarrow (sd_{nv_e,nd_e} = ed_{nv_e,nd_e} = SV_{nv_e} + DV_{nv_e}) \end{array}$$

Contraintes entre fenêtres de télédéchargement Les fenêtres de télédéchargement sont ordonnées à l'intérieur d'une fenêtre de visibilité, et si l'une d'entre elles est vide, alors les suivantes le sont aussi :

$$\begin{array}{ll} \forall k \in [1..NV], \forall l \in [1..MW_{max} - 1], & (ed_{k,l} \leq sd_{k,l+1}) \\ & (sd_{k,l} = ed_{k,l}) \rightarrow (sd_{k,l+1} = ed_{k,l+1}) \end{array}$$

Dates de début et de fin d'un télédéchargement La date de fin est la date de début augmentée de la durée de télédéchargement des observations :

$$\forall k \in [1..NV], \forall l \in [1..NW_{max}], ed_{k,l} = sd_{k,l} + \sum_{c \in OC, e \in OE_c} ((nv_e = k) \wedge (nd_e = l)) \cdot DDO_e$$

Contraintes entre observation et télédéchargement Si une observation n'est pas réalisée, elle n'est pas télédéchargée ; son télédéchargement débute après la fin de sa réalisation :

$$\begin{array}{ll} \forall e \in OE, & (no_e = 0) \rightarrow (nv_e = nd_e = 0) \\ & (nd_e \neq 0) \rightarrow (eo_e \leq sd_{nv_e,nd_e}) \end{array}$$

Séquentialité des actions Deux actions d'observation ou de télédéchargement ne peuvent pas s'exécuter en parallèle. Leurs fenêtres de réalisation sont donc disjointes :

$$\begin{array}{ll} \forall e_1, e_2 \in OE, & [so_{e_1}..eo_{e_1}] \cap [so_{e_2}..eo_{e_2}] = \emptyset \\ \forall k_1, k_2 \in [1..NV], \forall l_1, l_2 \in [1..NW_{max}], & [sd_{k_1,l_1}..ed_{k_1,l_1}] \cap [sd_{k_2,l_2}..ed_{k_2,l_2}] = \emptyset \\ \forall e \in OE, \forall k \in [1..NV], \forall l \in [1..NW_{max}], & [so_e..eo_e] \cap [sd_{k,l}..ed_{k,l}] = \emptyset \end{array}$$

Contraintes entre télédéchargement et station de réception Le télédéchargement d'une observation élémentaire ne peut s'effectuer que vers une station sol candidate de l'observation complexe correspondante :

$$\forall e \in OE, (nv_e \neq 0) \rightarrow (STATION_{nv_e} \in SS_{OBC_e})$$

État initial du satellite

$$\begin{array}{l} mm_1 = MM_0 \\ en_1 = EN_0 \\ er_1 = ER_0 \\ att_1 = ATT_0 \\ ti_1 = STA \end{array}$$

État initial des objectifs

$$\begin{array}{ll} \forall e \in OE, & ldetec_{e1} = LEDETEC_e \\ \forall e \in OER, & ((no_e = NO_e) \wedge (so_e = SO_e) \wedge (eo_e = SO_e + DDO_e) \wedge (a_e = A_e)) \\ \forall e \in OET, & ((nv_e = NV_e) \wedge (nd_e = ND_e) \wedge (ed_e = ED_e)) \end{array}$$

Évolution du niveau de mémoire libre La mémoire disponible est égale à la mémoire précédemment disponible, augmentée de ce qui est produit par les télédéchargements qui se terminent et diminuée de ce qui est consommé par les observations qui débutent :

$$\begin{array}{lll} \forall i \in [1..ns(tl)-1], mm_{i+1} = mm_i & - & (so_{obsc_{i+1}} = ti_{i+1}) \cdot SZ_{obsc_{i+1}} \\ & & + \sum_{e \in OE} (ed_{nv_e,nd_e} = ti_{i+1}) \cdot SZ_e \end{array}$$

Évolution du niveau d'énergie Le niveau d'énergie disponible est égal au niveau d'énergie précédemment disponible, augmenté de ce qui a été produit par les panneaux solaires si le satellite n'était pas en éclipse et diminué de ce qui a été consommé par la plate-forme, par l'instrument d'observation si le satellite était en observation, par l'antenne

de télédéchargement s'il était en télédéchargement, et par l'instrument de détection s'il était en détection de la couverture nuageuse, avec un maximum égal à EN_{max} :

$$\begin{array}{l}\forall i \in [1..ns(tl)-1], \\ en_{i+1} \quad = \quad min(EN_{max}, en_i \\ \qquad\qquad +(ti_{i+1}-ti_i)\cdot puissance(trajectoire(ti_i, ti_{i+1}, att_i.att_{i+1}, ac_i), ti_i) \\ \qquad\qquad -(ti_{i+1}-ti_i)\cdot((ac_i = OB)\cdot P_{ob} + (ac_i = DL)\cdot P_{dl} \\ \qquad\qquad +(ac_i = DETEC)\cdot P_{detec} + P_{sat}))\end{array}$$

Évolution du niveau d'ergol Le niveau d'ergol disponible est égal au niveau d'ergol précédemment disponible, diminué de ce qui a été consommé par les tuyères lors des actions de manœuvre orbitale :

$$\forall i \in [1..ns(tl)-1], er_{i+1} = er_i - (SM_{m_{i+1}} = ti_{i+1})\cdot CER_{m_{i+1}}$$

Conditions et effets d'une action d'observation

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], (ac_i = OB) \rightarrow & (so_{obsc_i} = ti_i) \\ & \wedge(eo_{obsc_i} = ti_{i+1}) \\ & \wedge(no_{obsc_i} \neq 0) \\ & \wedge(att_i = startObservationAttitude(obsc_i, ti_i)) \\ & \wedge respect(CONS_{obsc_i}) \\ & \wedge respect(CONS_{OBC_{obsc_i}}) \\ & \wedge(att_{i+1} = endObservationAttitude(obsc_i, ti_{i+1}))\end{array}$$

Conditions et effets d'une action de télédéchargement

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], & \\ (ac_i = DL) \rightarrow & (att_i = downloadAttitude(st_i, ti_i)) \\ & (att_{i+1} = downloadAttitude(st_i, ti_{i+1})) \\ & \wedge \bigvee_{k\in[1..NV], l\in[1..NW_{max}]} \left(\begin{array}{l}(sd_{k,l} = ti_i) \wedge (ed_{k,l} = ti_{i+1}) \\ \wedge(sd_{k,l} < ed_{k,l}) \wedge (STATION_k = st_i)\end{array} \right)\end{array}$$

Conditions et effets d'une action de détection

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], (ac_i = DETEC) \rightarrow & (att_i = detectionAttitude(ti_i)) \\ & \wedge(ti_{i+1} = ti_i + DD) \\ & \wedge(att_{i+1} = detectionAttitude(ti_{i+1})) \\ & \wedge\left(\bigwedge_{e\in detectedObs(ti_i)} (ldetec_{e_{i+1}} = ti_{i+1}) \right.\end{array}$$

Conditions et effets d'une action de rechargement des batteries

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], & \\ (ac_i = SUN) \rightarrow & (att_i = sunPointingAttitude(ti_i)) \\ & \wedge(\bigwedge_{k\in[1..NE]} ([ti_i..ti_i + d_i] \cap [SE_k..SE_k + DE_k] = \emptyset)) \\ & \wedge(ti_{i+1} = ti_i + d_i) \\ & \wedge(att_{i+1} = sunPointingAttitude(ti_{i+1}))\end{array}$$

Conditions et effets d'une action de pointage géocentrique

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], (ac_i = GEO) \rightarrow & (att_i = geoPointingAttitude(ti_i)) \\ & \wedge(ti_{i+1} = ti_i + d_i) \\ & \wedge(att_{i+1} = geoPointingAttitude(ti_{i+1}))\end{array}$$

Conditions et effets d'une action de rendez-vous en attitude

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], (ac_i = RDV) \rightarrow & (ti_{i+1} = ti_i + d_i) \\ & \wedge(d_i \geq minDur(att_i, targ\_att_i)) \\ & \wedge(att_{i+1} = targ\_att_i)\end{array}$$

Conditions et effets d'une action de manœuvre orbitale

$$\begin{array}{ll}\forall i \in [1..ns(tl)-1], (ac_i = MAN) \rightarrow & (ti_i = SM_{m_i}) \\ & \wedge(att_i = SATT_{m_i}) \\ & \wedge(ti_{i+1} = ti_i + DM_{m_i}) \\ & \wedge(att_{i+1} = EATT_{m_i})\end{array}$$

Conditions entre attributs

$$\begin{array}{ll}\forall i \in [1..ns(tl)], & (obsc_i \neq \perp) \leftrightarrow (ac_i = OB) \\ & (st_i \neq \perp) \leftrightarrow (ac_i = DL) \\ & (d_i \neq 0) \leftrightarrow (ac_i \in \{SUN, GEO, RDV\}) \\ & (targ\_att_i \neq \perp) \leftrightarrow (ac_i = RDV)\end{array}$$

On impose au satellite d'effectuer un pointage géocentrique entre les dates END et END_{ec} afin de s'assurer qu'il conserve suffisamment d'énergie pour pouvoir ressortir sans problème de la période d'éclipse à la date END_{ec} :

$$\begin{array}{ll}t_{ns(tl)} = END_{ec} & \\ (ti_i \geq END) \rightarrow (i \geq ns(tl)-2) & \\ (ac_{ns(tl)-1} = GEO) & \wedge(ac_{ns(tl)-2} = RDV) \\ & \wedge(d_{ns(tl)-2} = minDur(att_{ns(tl)-2}, geoPointingAttitude(ti_{ns(tl)-1}))) \\ & \wedge(targ\_att_{ns(tl)-2} = geoPointingAttitude(ti_{ns(tl)-1}))\end{array}$$

Conditions entre variables classiques et attributs

$$\left(\bigvee_{e\in OE_c|c\in OC}(no_e \neq 0)\right) \leftrightarrow \left(\bigvee_{i\in[1..ns(tl)]}(ac_i = OB)\right)$$
$$\left(\bigvee_{e\in OE_c|c\in OC}(nd_e \neq 0)\right) \leftrightarrow \left(\bigvee_{i\in[1..ns(tl)]}(ac_i = DL)\right)$$

La figure 3.3 représente une instanciation de la timeline avec 15 étapes, sur l'horizon de planification.

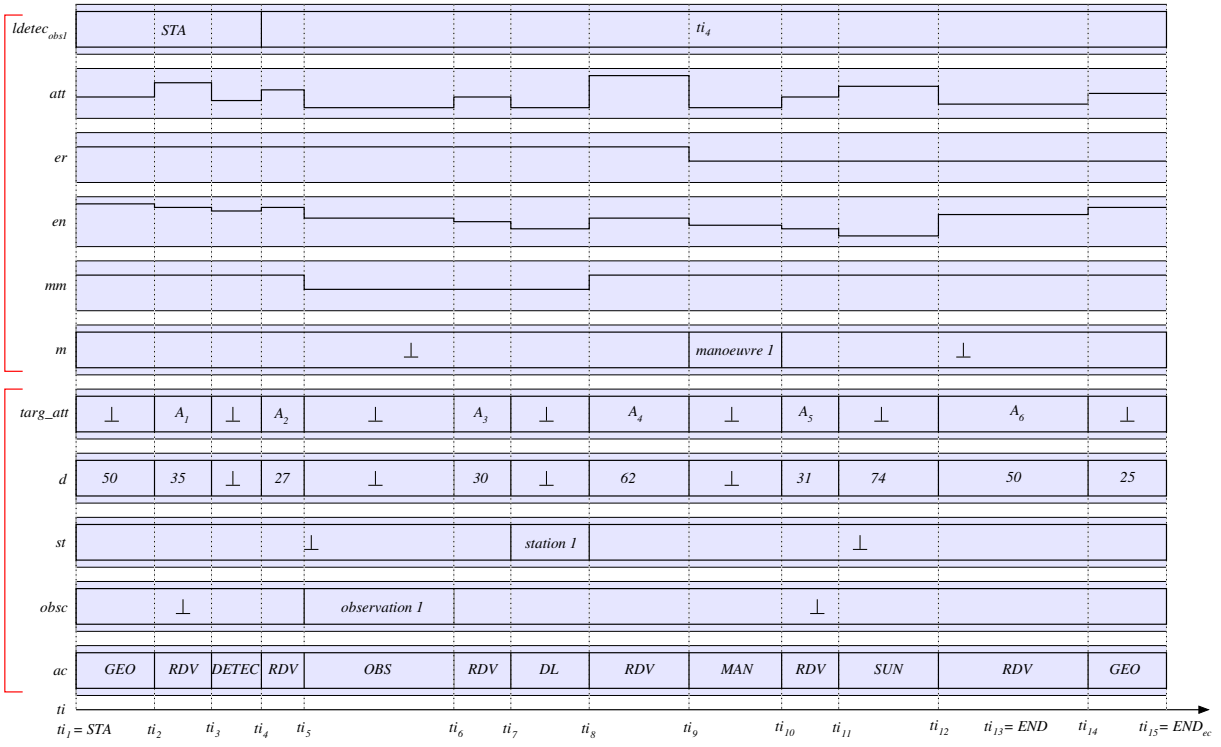


FIG. 3.3: Exemple de timeline

3 Évaluation et critère d'optimisation

Au cours d'une telle mission d'observation, on peut vouloir chercher à optimiser différents critères pouvant être parfois antinomiques. Un organisme civil pourra préférer la réalisation complète plutôt que partielle des observations complexes, ou maximiser la

qualité des prises de vues qu'il demande. Cela reviendra à minimiser la couverture nuageuse au dessus des zones imagées et à faire tendre l'angle de prise de vue vers 0 ° (satellite à la verticale de la zone). De son côté, un organisme militaire pourra souhaiter obtenir une image le plus rapidement possible, ce qui reviendra à minimiser le délai de livraison. Enfin, les opérateurs en charge du maintien à poste du satellite souhaiteront éviter de trop puiser dans les réserves en énergie du satellite. Cela signifiera privilégier les périodes de recharge des batteries.

Évaluer la valeur d'un plan du satellite sur un certain horizon temporel $[STA..END]$ revient à déterminer la valeur de l'état final de ce plan (l'état dans lequel se trouve le système à la date END). On cherchera donc à évaluer :

- d'une part le gain réalisé par la mission jusque dans cet état final : cela correspond à sommer la valeur de chacune des observations réalisées ou télédéchargées, pondérées par un poids fonction de leurs priorités ;
- et d'autre part l'espérance de gain que le système peut avoir sur le reste de la vie du satellite. Un moyen simple et cohérent sera d'évaluer le niveau d'énergie restant dans les batteries du satellite à la date END.

Dans cette section nous définissons une fonction d'évaluation des objectifs réalisés de la mission et une fonction d'évaluation de la quantité d'énergie restante dans les batteries du satellite. À partir de ces deux fonctions nous définissons le critère à optimiser tout au long de la mission.

3.1 Définitions

Pour chaque observation complexe $c \in OC$, on définit :

- sa date de fin de télédéchargement $endD_c = \max_{e \in OE_c} \{ed_{nd_e}\}$,
- son pourcentage de télédéchargement (nombre d'observations élémentaires télédéchargées sur le nombre total d'observations élémentaires composant l'observation complexe) :

$$pTel_c = \frac{\sum_{e \in OE_c} (nd_e \neq 0)}{|OE_c|}$$

- son pourcentage de réalisation (nombre d'observations élémentaires télédéchargées ou réalisées sur le nombre total d'observations élémentaires composant l'observation complexe) :

$$pRea_c = \frac{\sum_{e \in OE_c} (no_e \neq 0)}{|OE_c|}$$

Pour chaque observation complexe c et chaque observation élémentaire $e \in OE_c$, on définit également :

– son angle d'observation $a_e \in [-\frac{\pi}{2}..\frac{\pi}{2}]$, fonction de l'observation e et de sa date de début de réalisation so_e : $a_e = observationAngle(e, so_e)$
– la fraîcheur de son information de détection, fonction de la date de dernière détection de sa couverture nuageuse (voir figure 3.4) :

$$f_e(ldetec_e) = \begin{cases} 1 \text{ si } so_e - TN \leq ldetec_e \leq so_e \\ 0.8 \text{ si } so_e - 2TN \leq ldetec_e \leq so_e - TN \\ 0.5 \text{ sinon} \end{cases}$$

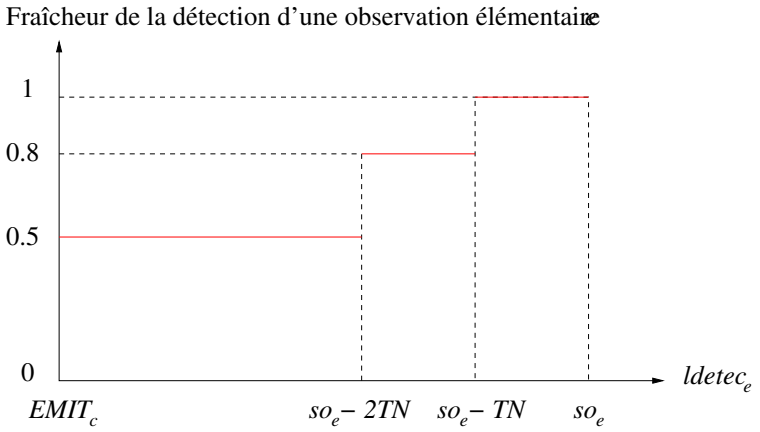


FIG. 3.4: Fraîcheur de la détection

– sa vitesse de livraison, fonction de sa date de fin de télédéchargement et de la date d'émission de l'observation complexe associée (voir figure 3.5) :

$$liv_e(ed_e) = \begin{cases} \frac{1}{2}(\frac{ed_e^2 - DEAD_c^2}{EMIT_c^2 - DEAD_c^2}) + \frac{1}{2} \text{ si } EMIT_c \leq ed_e \leq DEAD_c \\ 0 \text{ sinon} \end{cases}$$

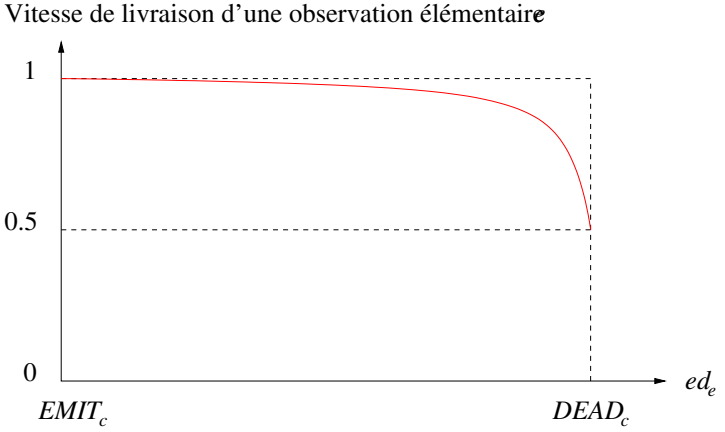


FIG. 3.5: Vitesse de livraison

3.2 Évaluation d'une observation complexe

La valeur de l'état d'une observation complexe c, notée $v(c)$, est fonction :

- de sa priorité,
- de son pourcentage de télédéchargement,
- de son pourcentage de réalisation,
- du pourcentage de couverture nuageuse au dessus de chaque observation élémentaire réalisée,
- de la fraîcheur de l'information de détection pour chacune de ses observations élémentaires,
- de l'angle de prise de vue lors de la réalisation de chacune de ses observations élémentaires,
- et de la date de livraison de chacune de ses observations élémentaires.

La priorité, le pourcentage de télédéchargement et le pourcentage de réalisation sont des paramètres d'une observation complexe. En revanche, la couverture nuageuse, la fraîcheur de l'information de détection, l'angle de prise de vue et la date de livraison sont des paramètres d'une observation élémentaire. Pour pouvoir les évaluer au niveau d'une observation complexe, nous avons choisi d'en faire la moyenne sur toutes les observations élémentaires d'une observation complexe.

Nous définissons ainsi la couverture nuageuse moyenne d'une observation complexe c : $CN_c = \frac{1}{|OE_c|} \sum_{e \in OE_c} CN_e$ et son taux de non couverture nuageuse : $\overline{cn}_c = 1 - CN_c$
la fraîcheur moyenne de son information de détection : $f_c = \frac{1}{|OE_c|} \sum_{e \in OE_c} f_e(ldetec_e)$
son angle de prise de vue moyen : $a_c = \frac{1}{|OE_c|} \sum_{e \in OE_c} a_e$
et sa vitesse de livraison moyenne : $liv_c = \frac{1}{|OE_c|} \sum_{e \in OE_c} liv_e$

La date de télédéchargement d'une observation élémentaire réalisée mais non encore télédéchargée, est fixée de façon optimiste à la date de fin de la prochaine fenêtre de visibilité d'une station sol pouvant recevoir l'observation en question (on suppose que l'observation élémentaire sera télédéchargée au cours de cette fenêtre de visibilité).

La priorité d'une observation complexe est un entier compris entre 1 et 3, où 3 est la priorité maximale. Afin de privilégier une observation de priorité p par rapport à tout ensemble d'observations de priorités inférieures, on associe à chaque priorité $p \in \{1, 2, 3\}$ un poids w_p tel que : $w_1 = 1$ et $\forall p \in \{2, 3\}, w_p = N_{p-1} \cdot w_{p-1} + 1$ où N_{p-1} est le nombre maximal d'observations complexes de priorité $p - 1$.

On définit alors la valeur d'une observation complexe par[1] :

$$v(c) = w_{PRIO_c} \cdot \frac{1}{2}(pTel_c + pRea_c) \cdot \overline{cn}_c \cdot f_c \cdot a_c \cdot liv_c$$

3.3 Évaluation du niveau d'énergie restante

Nous définissons la valeur du niveau d'énergie restante $en \in [EN_{min}..EN_{max}]$ de la façon suivante (voir figure 3.6) :

$$v(en) = \sqrt{\frac{en - EN_{min}}{EN_{max} - EN_{min}}}$$

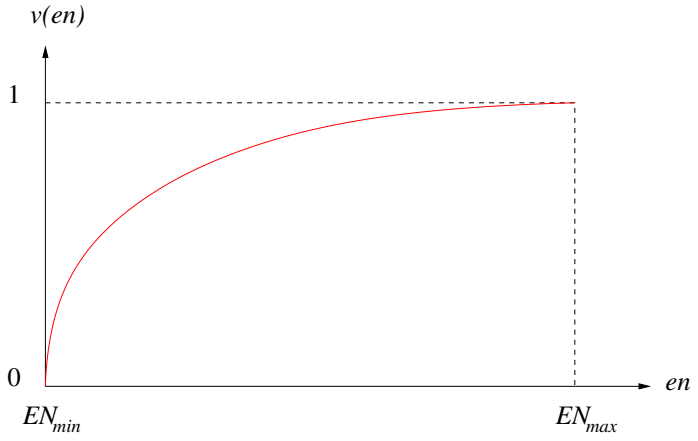


FIG. 3.6: Valeur du niveau d'énergie restante

3.4 Critère d'optimisation

Le critère à optimiser sur l'horizon de planification $[STA..END]$ sera la valeur du plan d'activités du satellite sur cet horizon, c'est-à-dire la valeur de l'état du système à la fin de cet horizon.

Afin de maximiser la valeur de l'état E du système à la date END, on cherche à maximiser la valeur de l'état des observations complexes connues dans l'état E : $\sum_{c \in OC} v(c)$. Puis dans un second temps, on cherche à maximiser l'espérance de gain futur, c'est-à-dire le gain auquel le satellite pourra prétendre entre l'état E et la fin de sa mission. Pour

[1] Le terme $\frac{1}{2}(pTel_c + pRea_c)$ a été préféré à $pTel_c \times pRea_c$ pour pouvoir évaluer différemment deux observations complexes dont aucune observation élémentaire n'aurait encore été télédéchargée ($pTel_c = 0$).

cela, on suppose qu'il pourra réaliser d'autant plus d'observations qu'il aura un niveau d'énergie important dans l'état E. Ainsi, on cherchera à maximiser la valeur du niveau d'énergie restante dans cet état : $v(en_{ns(tl)})$.

Exemple 3 *Soient deux plans d'activités P_1 et P_2 sur l'horizon $[STA..END]$, amenant le satellite respectivement dans les états E_1 (avec un ensemble d'observations complexes d'état OC_1 et un niveau d'énergie en_1) et E_2 (avec un ensemble d'observations complexes d'état OC_2 et un niveau d'énergie en_2).*

- *P_1 sera jugé meilleur que P_2 si $\sum_{c \in OC_1} v(c) > \sum_{c \in OC_2} v(c)$, et inversement.*
- *Si $\sum_{c \in OC_1} v(c) = \sum_{c \in OC_2} v(c)$, alors P_1 sera jugé meilleur que P_2 si et seulement si $v(en_1) > v(en_2)$.*
- *Si $\sum_{c \in OC_1} v(c) = \sum_{c \in OC_2} v(c)$ et $v(en_1) = v(en_2)$ les deux plans seront jugés équivalents.*

Une partie de ces travaux a donné lieu à une communication nationale [Beaumet *et al.*, 2006] aux Journées Francophones Planification, Décision, Apprentissage pour la conduite de systèmes (JFPDA'06) et une communication internationale [Beaumet, 2006] au "doctoral consortium" de la conférence ICAPS'06 (International Conference on Automated Planning and Scheduling).

Chapitre 4

Agilité - Guidage en attitude

1 Notions de mécanique spatiale

1.1 Référentiels

1.1.1 Quelques définitions

La Terre En première approximation, nous supposerons que la Terre est un corps parfaitement sphérique, solide et rigide. Mais il faut garder à l'esprit que ceci est une simplification grossière : sa forme n'est ni une sphère, ni un ellipsoïde parfait ; son centre de masse n'est pas confondu avec son centre géométrique ; le mouvement du centre de la Terre est différent de celui du barycentre du système Terre - Lune ; la rotation de la croûte terrestre est légèrement différente de la rotation d'ensemble de la planète, estimée à partir de son moment cinétique résultant. Dans certains cas nous serons donc amenés à préciser le modèle que nous utiliserons, notamment pour le calcul des attitudes du satellite en pointage station, ou en observation.

Sphère céleste La sphère céleste est une sphère imaginaire de rayon quelconque et dont le centre est occupé par la Terre. Ce concept astronomique, hérité de l'antiquité, permet de représenter tous les astres tels qu'on les voit depuis la Terre. C'est le seul élément considéré comme fixe et dans lequel les directions des étoiles lointaines ont des positions fixes.

Pôles célestes Les deux pôles célestes sont les points de la sphère céleste vers lesquels pointe l'axe de rotation de la Terre et autour desquels le ciel semble donc tourner. À cause de la précession des équinoxes, l'axe de rotation de la Terre n'est pas fixe avec le temps, ainsi la position des deux pôles célestes varie au cours du temps. Par exemple, il y a environ 14 000 ans, c'était l'étoile Véga qui déterminait le pôle nord céleste, et ce sera à nouveau le cas dans environ 12 000 ans.

Équateur céleste C'est le plan perpendiculaire à l'axe des pôles célestes. C'est celui que nous appelons communément l'équateur terrestre.

Plan de l'écliptique C'est le plan moyen de l'orbite terrestre, ou plus exactement du système Terre - Lune. Ce plan varie avec un mouvement régulier à long terme et un mouvement harmonique de faible amplitude à court terme. Le plan écliptique moyen considéré varie avec le mouvement à long terme, mais moyenne le mouvement harmonique.

Point Vernal Noté γ, le point Vernal est dans la direction Terre - Soleil au moment de l'équinoxe de printemps. C'est un des deux points d'intersection de l'équateur céleste et du plan de l'écliptique.

Nœud ascendant Le nœud ascendant est le point de l'orbite d'un satellite où celui-ci traverse l'écliptique depuis l'hémisphère céleste Sud vers l'hémisphère Nord.

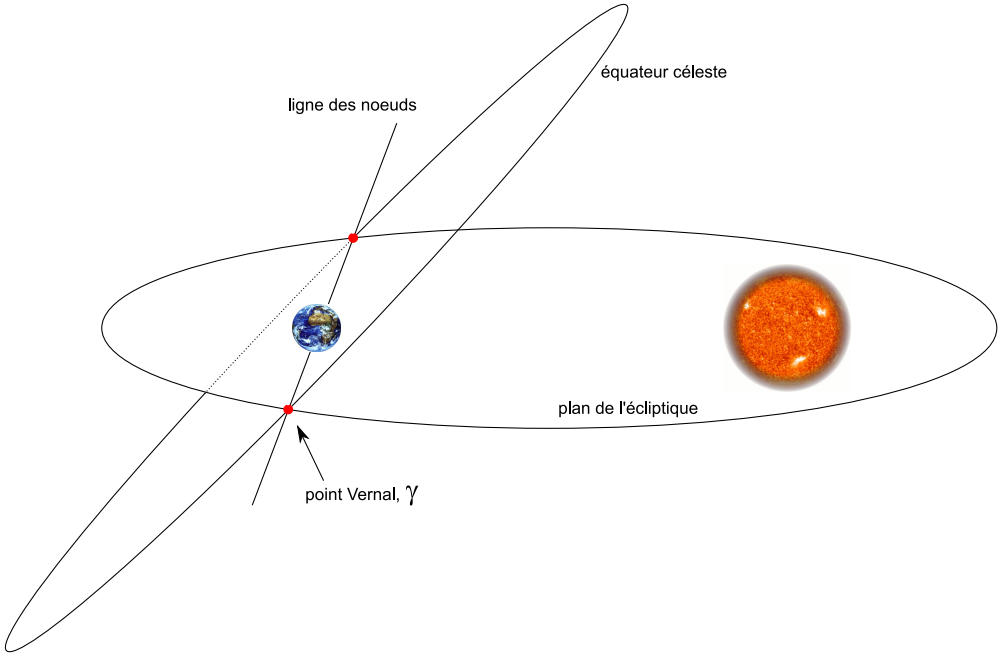


FIG. 4.1: Vocabulaire de la mécanique céleste

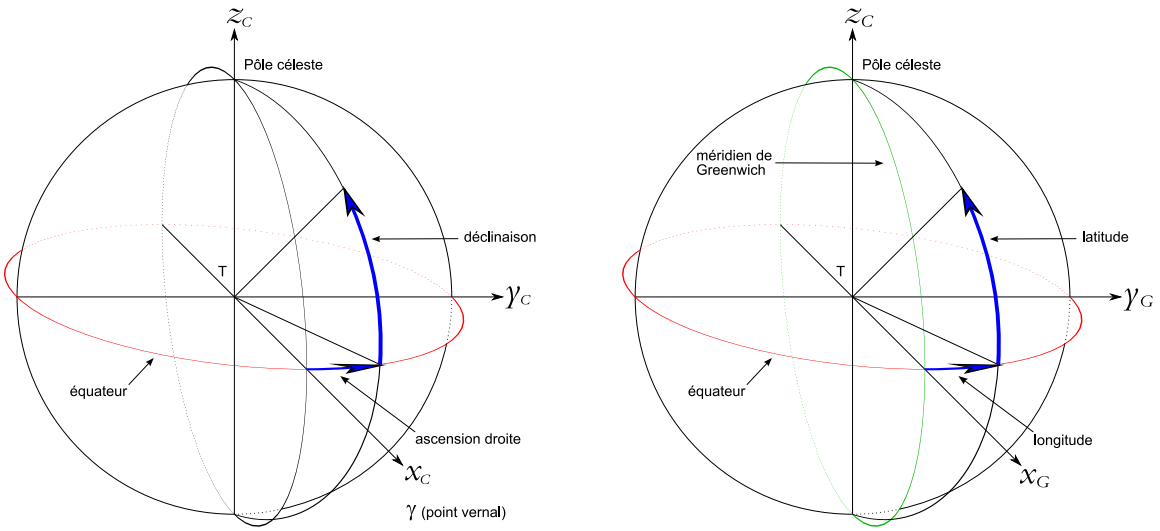
1.1.2 Repères et systèmes de coordonnées

Dans ce chapitre nous utiliserons principalement les repères de référence suivants, couramment utilisés en mécanique spatiale :

Repère inertiel Le repère équatorial terrestre $\mathcal{R}_C$ (aussi appelé repère céleste) est un repère supposé fixe, qui a son origine au centre de la Terre. Son axe $\vec{z}_C$ est dirigé vers le pôle céleste boréal P_C et son axe $\vec{x}_C$ est dirigé vers le point γ. Les coordonnées sphériques d'une direction dans ce repère sont l'*ascension droite* α (traditionnellement exprimée en unités de temps de 0 à 24 heures) et la *déclinaison* δ (traditionnellement exprimée en degrés entre $\pm 90^\circ$).

Repère géographique Lié à la Terre, le repère géographique $\mathcal{R}_G$ est positionné au centre de la Terre. Son axe $\vec{z}_G$ est dirigé vers le pôle céleste boréal P_C et son axe $\vec{x}_G$ est dans le plan du méridien passant par Greenwich. La direction d'un point y est repérée par sa *longitude* λ et sa latitude ϕ.

Les repères céleste et géographique sont représentés sur la figure 4.2.



(a) Repère céleste (b) Repère géographique

FIG. 4.2: Repères céleste et géographique

Repère orbital local Le repère orbital local $\mathcal{R}_L$, d'origine le centre de masse du satellite, permet de repérer la position orbitale du satellite autour de la Terre. Son axe $\vec{z}_L$ est pris selon la direction Terre - Satellite, et son axe $\vec{y}_L$ selon la direction du moment cinétique moyen du satellite sur son orbite[1].

[1]Une autre convention, parfois utilisée, préconise de choisir $\vec{y}_L$ dans la direction du vecteur vitesse du satellite. Mais elle présente l'inconvénient d'orienter le vecteur $\vec{x}_L$ dans le sens opposé au moment cinétique moyen du satellite, et donc de compter la vitesse orbitale du satellite négativement.

Repère zone Pour tout point M à la surface de la Terre, on définit le repère zone $\mathcal{R}_A$ dont l'origine est positionnée en M. Son axe $\vec{z}_A$ est dirigé vers le zénith et son axe $\vec{x}_A$ dans la direction d'observation de la zone associée au point M.

Les repères zone et orbital local sont représentés sur la figure 4.3.

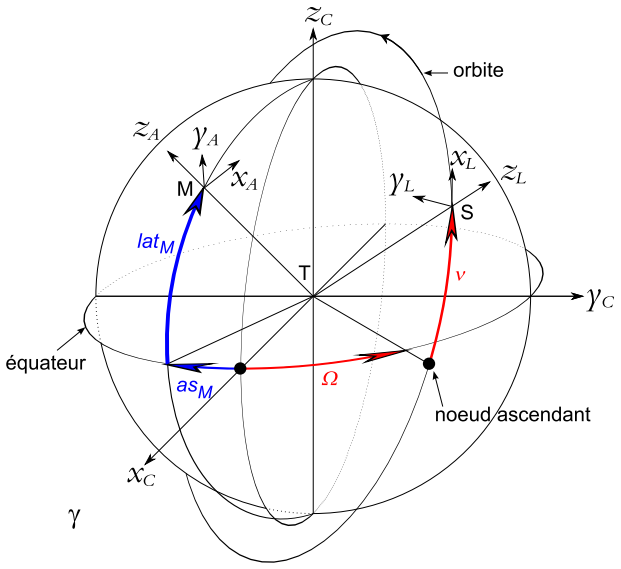


FIG. 4.3: Repères zone et orbital local

Repère satellite D'origine le centre du satellite, le repère $\mathcal{R}_S$ lié au satellite permet de définir son attitude, c'est-à-dire son orientation par rapport à son centre de masse. Ses axes correspondent à des axes privilégiés du satellite : l'axe $\vec{z}_S$ coïncide avec l'axe de l'instrument d'observation du satellite et est de sens opposé au sens de visée, tandis que les axes $\vec{x}_S$ et $\vec{y}_S$ complètent le repère dans un plan parallèle au plan des panneaux solaires du satellite (cf. figure 4.4).

1.1.3 Origine des temps

Dans tout le document, nous avons choisi comme origine des temps le 1er janvier 2000 à 12 heures. En fonction de sa date d, exprimée en jours juliens[2], l'instant t d'un évènement sera donné en secondes par :

$$t = 86400(d - 2451545)$$

[2]Les jours juliens sont un système de datation continu, dérivé d'un système proposé par Joseph-Juste Scaliger en 1583. Ce système compte le temps écoulé, en jours décimaux, depuis le lundi 1er janvier -4172 (c'est-à-dire le 1er janvier 4173 avant JC) à midi. Ainsi, la date du 2 janvier -4172 à minuit vaut $JJ = 0.5$, et celle du 1er janvier 2000 à midi vaut $J2000 = 2451545.0$.

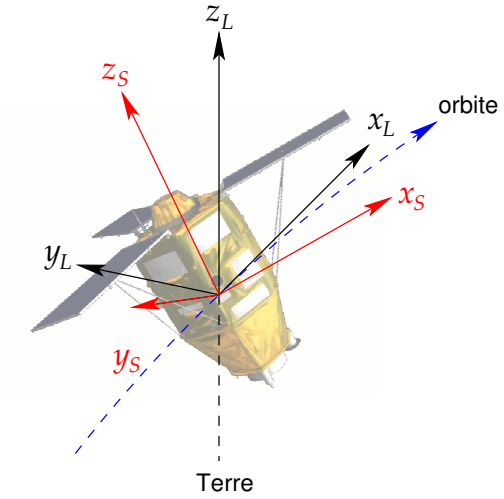


FIG. 4.4: Repères satellite et orbital local

1.1.4 Quelques notations

On note s_α et c_α respectivement le sinus et le cosinus de l'angle α.
$R_\psi(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha \\ 0 & s_\alpha & c_\alpha \end{pmatrix}$ désigne la matrice de rotation d'un angle α autour du premier vecteur de base,
$R_\theta(\alpha) = \begin{pmatrix} c_\alpha & 0 & s_\alpha \\ 0 & 1 & 0 \\ -s_\alpha & 0 & c_\alpha \end{pmatrix}$ la matrice de rotation d'un angle α autour du deuxième vecteur de base, et
$R_\phi(\alpha) = \begin{pmatrix} c_\alpha & -s_\alpha & 0 \\ s_\alpha & c_\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$ la matrice de rotation d'un angle α autour du troisième vecteur de base.
On note $\vec{u}^{R_i}$ le vecteur $\vec{u}$ exprimé dans le repère R_i. Lorsque le repère n'est pas précisé, c'est que le vecteur est exprimé dans le repère inertiel $\mathcal{R}_C$: $\vec{u} = \vec{u}^{\mathcal{R}_C}$. On note $\vec{u} \times \vec{v}$ le produit vectoriel de $\vec{u}$ par $\vec{v}$, et $\langle \vec{u} \cdot \vec{v} \rangle$ leur produit scalaire. $\vec{u}^T$ définit le vecteur transposé du vecteur $\vec{u}$. Enfin on note $\angle(\vec{u}, \vec{v})$ l'angle entre les vecteurs $\vec{u}$ et $\vec{v}$.

1.2 Cinématique des satellites

1.2.1 Éléments orbitaux

Une orbite elliptique peut se définir dans l'espace à l'aide de six paramètres permettant de calculer très précisément la trajectoire complète [CNES, 1998]. Deux de ces paramètres

(excentricité e et demi-grand axe a) définissent la trajectoire dans un plan, trois autres (inclinaison i, ascension droite du nœud ascendant Ω et argument du périgée ω) définissent l'orientation du plan dans l'espace et le dernier (instant de passage au périgée t_P) définit la position du satellite.

On peut associer au plan orbital d'un satellite S, deux repères $\mathcal{R}_N$, $\mathcal{R}_P$ d'origine le centre T de la Terre. Notons $\vec{x}_N$ le vecteur unitaire dirigé selon l'intersection du plan orbital avec le plan équatorial et orienté vers le nœud ascendant, $\vec{x}_P$ dirigé selon le vecteur $\overrightarrow{TP}$, le point P étant le périgée de l'orbite, et $\vec{z}_N = \vec{z}_P$ la normale au plan orbital, dirigé selon le moment cinétique $\vec{h} = \overrightarrow{TS} \times \vec{V}_S$ du satellite ($\vec{V}_S$ définissant la vitesse du satellite sur son orbite).

On a donc :

$$\begin{aligned} i &= \angle(\vec{z}_C, \vec{z}_N) \\ \Omega &= \angle(\vec{x}_\gamma, \vec{x}_N) \\ \omega &= \angle(\vec{x}_N, \vec{x}_P) \end{aligned}$$

1.2.2 Position et vitesse du satellite

En notant r le rayon de l'orbite du satellite, E l'anomalie excentrique du satellite S définie comme l'angle au centre repérant le point Q, projection orthogonale de S sur le cercle de rayon a, ν son anomalie vraie qui est l'argument de S dans le repère $\mathcal{R}_P$, n son mouvement moyen relié à la période τ de l'orbite par $n = \frac{2\pi}{\tau}$ et M son anomalie moyenne définie à tout instant t par $M = n(t - t_P)$, les coordonnées du satellite dans le repère $\mathcal{R}_P$ sont données par :

$$\left\{ \begin{array}{lcl} x &=& r \cdot \cos(\nu) = -a \cdot e + a \cdot \cos(E) = a(\cos(E) - e) \\ y &=& r \cdot \sin(\nu) = \frac{a \cdot \sqrt{1-e^2}}{a} a \cdot \sin(E) = a \cdot \sqrt{1 - e^2} \cdot \sin(E) \\ z &=& 0 \end{array} \right. \tag{4.1}$$

En dérivant les relations 4.1, on obtient la vitesse du satellite dans $\mathcal{R}_P$:

$$\left\{ \begin{array}{lcl} \dot{x} &=& \dot{r} \cdot \cos(\nu) - r \cdot \dot{\nu} \cdot \sin(\nu) = -a \cdot \dot{E} \cdot \sin(E) \\ \dot{y} &=& \dot{r} \cdot \sin(\nu) + r \cdot \dot{\nu} \cdot \cos(\nu) = a \cdot \dot{E} \sqrt{1 - e^2} \cos(E) \\ \dot{z} &=& 0 \end{array} \right. \tag{4.2}$$

où $\dot{E}$ s'obtient en dérivant l'équation de Kepler suivante :

$$M = E - e \cdot \sin(E)$$

$$\dot{E} = \frac{n}{1 - e \cdot \cos(E)}$$

2 Contrôle d'attitude

Le SCAO (Système de Contrôle d'Attitude et d'Orbite) est en charge du maintien du satellite sur son orbite, et du contrôle de son attitude. On appelle attitude (respectivement vitesse en attitude) d'un satellite son orientation (respectivement sa vitesse de rotation) autour de son centre d'inertie.

2.1 Contrôle d'attitude par roues à réaction

Cette méthode de contrôle d'attitude repose sur le principe de conservation du moment cinétique total du satellite. La figure 4.5 décrit le schéma de principe d'une roue à réaction, encore appelée roue à inertie. Un moteur d'axe $\vec{z}$ lié au bâti du satellite entraîne en rotation

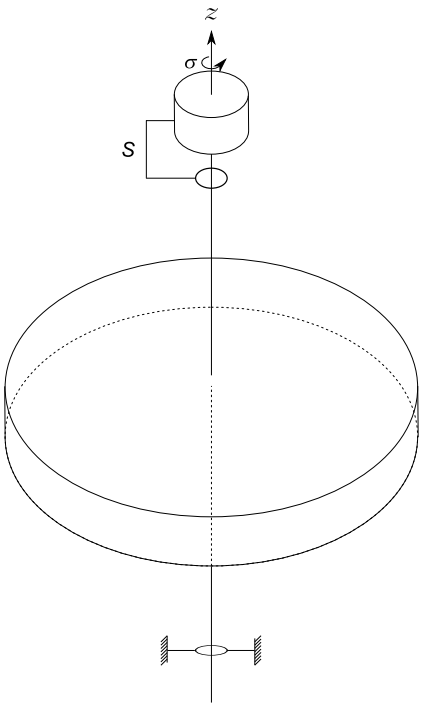


FIG. 4.5: Schéma de principe d'une roue à réaction

à la vitesse $\dot{\sigma}\vec{z}$ un axe à l'extrémité duquel est fixé un corps cylindrique de grande inertie, la roue à réaction. Si on note j le moment d'inertie de la roue à réaction autour de l'axe $\vec{z}$, son moment cinétique relativement au carter est donné par :

$$\vec{h} = j\dot{\sigma}\vec{z} = h\vec{z}$$

Comme le moment cinétique $\vec{H}$ du système complet (satellite et roue à réaction) reste constant (et même nul si le satellite était initialement immobile, $\vec{H} = \vec{0}$), le couple moteur résultant est donné par :

$$\vec{M} = \vec{H} - h\vec{z} = -h\vec{z}$$

Si on note J le moment d'inertie du satellite autour de l'axe z, sa vitesse $\dot{\Sigma}$ de rotation autour de cet axe est égale à :

$$\dot{\Sigma} = -\frac{j}{J}\dot{\sigma}$$

2.2 Contrôle d'attitude par actionneurs gyroscopiques

Une méthode de contrôle d'attitude utilisée jusqu'alors sur les gros véhicules spatiaux, est en train de se généraliser pour les engins plus petits et nécessitant des mouvements en attitude précis et rapides : le contrôle d'attitude par actionneurs gyroscopiques [Margulies et Aubrun, 1978]. Ce système de contrôle d'attitude repose également sur le principe d'échange de moment cinétique entre un système interne commandé (une grappe d'actionneurs gyroscopiques) et le satellite.

2.2.1 Principe des gyrodynes un axe

La figure 4.6 décrit le schéma de principe d'un actionneur gyroscopique 1-axe, appelé aussi gyrodyne ou CMG (Control Moment Gyro).

Un moteur d'axe $\vec{z}$, lié au bâti du satellite, fait précessionner un carter contenant une toupie. L'axe $\vec{z}$ est fixe dans le repère satellite. Si on note σ l'angle de précession imposé par le moteur de précession, l'axe $\vec{x}$ commun au carter et à la toupie tourne relativement au satellite à la vitesse $\dot{\sigma}\vec{z}$.

Le moteur de rotation propre de la toupie entraîne celle-ci à une vitesse de rotation $\dot{\phi}\vec{x}$ relativement au carter. Si on note j le moment d'inertie de la toupie autour de l'axe $\vec{x}$, son moment cinétique relativement au carter est donné par :

$$\vec{h} = j\dot{\phi}\vec{x} = h\vec{x}$$

avec $h = j\dot{\phi} \simeq$ cte.

Le couple gyroscopique moteur résultant de ce moment cinétique est donné par :

$$\begin{aligned} \vec{M} &= \dot{\sigma}\vec{z} \times \vec{h} = \dot{\sigma}\vec{z} \times h\vec{x} \\ \vec{M} &= h\dot{\sigma}\vec{y} \end{aligned}$$

Ce moment est transmis au carter par les paliers notés B et B', et transmis du carter au bâti du satellite par les paliers notés A et A' sur la figure 4.6.

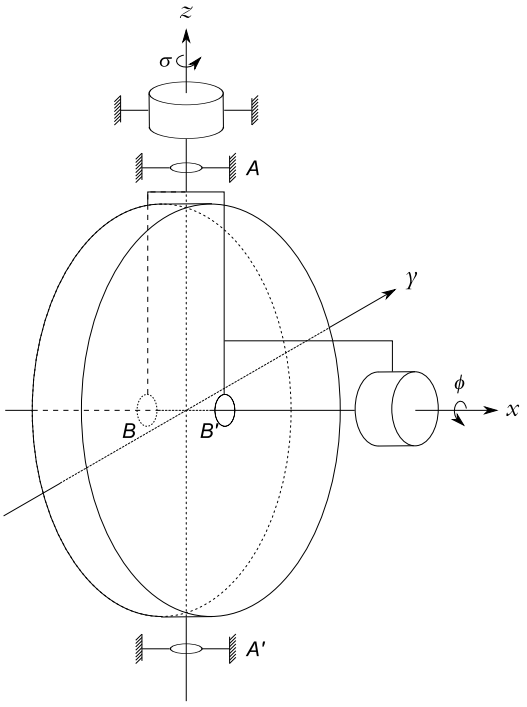


FIG. 4.6: Schéma de principe d'un actionneur gyroscopique 1-axe

2.2.2 Intérêt par rapport aux roues à inertie

Considérons un satellite que l'on souhaite mettre en rotation autour d'un axe $\vec{u}$ passant par son centre de masse. Soit $I = 100$ kg.m^2 son moment d'inertie autour de $\vec{u}$. C'est par exemple un cylindre de 200 kg et de 1 m de rayon.

Considérons une rotation de ce satellite consistant à l'amener au bout de $T = 1$ seconde avec une accélération de rotation constante $\dot{\Omega}_{max} = 1$ rd.s^{-2}, d'une vitesse nulle à une vitesse $\Omega_{max} = 1$ rd.s^{-1}. Le couple moteur maximal C_{max}, nécessaire à cette rotation est :

$$C_{max} = I \cdot \dot{\Omega}_{max} = 100 \text{ N.m}$$

la puissance maximale Π_{max}, nécessaire :

$$\Pi_{max} = C_{max} \cdot \Omega_{max} = 100 \text{ W}$$

et l'énergie utile E, nécessaire :

$$E = \frac{1}{2} I \cdot \Omega_{max}^2 = 50 \text{ J}$$

Exemple avec des roues à inertie Supposons que cette rotation soit contrôlée par une roue à inertie de même axe $\vec{u}$ et de moment d'inertie $j = 1$ kg.m^2 autour de cet axe. C'est par exemple un cylindre de 8 kg et de 50 cm de rayon. Pour un contrôle trois axes, la masse totale des roues à inertie représenterait 24 kg, et par exemple 35 kg avec les carters, moteurs, paliers, ..., soit 35 kg de motorisation en attitude pour 165 kg restants.

Mais dans ce mouvement, la roue à inertie est accélérée avec une accélération :

$$\dot{\omega}_{max} = \frac{1}{j} C_{max} = 100 \text{ rd.s}^{-2}$$

et atteint une vitesse :

$$\omega_{max} = 100 \text{ rd.s}^{-1}$$

Le moteur fournit donc une puissance maximale :

$$\pi_{max} = C_{max} \cdot \omega_{max} = 100 \times 100 = 10 \text{ kW}$$

et dépense une énergie :

$$e = \frac{1}{2} j \cdot \omega_{max}^2 = 5000 \text{ J}$$

On dépense donc $100 = \frac{I}{j}$ fois plus d'énergie inutile que d'énergie utile.

Exemple avec des gyrodynes Considérons maintenant une motorisation à l'aide de deux gyrodynes d'axe de précession commun orthogonal à $\vec{u}$. Les moments cinétiques $\vec{h}_1$ et $\vec{h}_2$ des deux toupies sont égaux, opposés et orthogonaux à $\vec{u}$ à l'instant initial.

$$h = |\vec{h}_1| = |\vec{h}_2| = \text{cte}$$

Supposons que Ω_{max} soit atteint en faisant précessionner les deux gyrodynes en sens inverse (de manière à aligner $\vec{h}_1 + \vec{h}_2$ avec $\vec{u}$), en une seconde, en faisant passer la somme $\vec{h} = \vec{h}_1 + \vec{h}_2$ de $\vec{0}$ pour $\sigma = 0$ à $\vec{h} = \vec{h}_1 + \vec{h}_2 = I \cdot \Omega_{max} \vec{u}$ pour $\sigma_{max} = \frac{\pi}{4}$ rd. Il en résulte que :

$$2h \cdot \sin(\sigma_{max}) = h\sqrt{2} = I \cdot \Omega_{max} = 100$$

soit :

$$h = \frac{100}{\sqrt{2}} = 70.7 \text{ N.m.s}$$

Ce qui peut être réalisé avec une toupie ayant un moment d'inertie $j_T = \frac{1}{10\sqrt{2}} = 0.0707$ kg.m^2 et tournant à une vitesse de 1000 rd.s^{-1}. Une toupie de ce type peut être obtenue avec une couronne de 1 kg et 0.266 m de rayon, soit 2 kg pour les deux toupies. On constate alors que des performances identiques peuvent être

obtenues avec des gyrodynes de masse et d'encombrement largement inférieurs à ceux des roues à inertie.

Supposons que la rotation en précession soit de la forme :

$$\begin{aligned} \sigma &= \frac{\pi}{4}t^2(3-2t) \\ \dot{\sigma} &= \frac{3\pi}{2}t(1-t) \\ \ddot{\sigma} &= \frac{3\pi}{2}(2t-1) \end{aligned}$$

L'angle σ passe bien de 0 à $\frac{\pi}{4}$ en une seconde, en partant d'une vitesse nulle pour arriver en $\sigma = \frac{\pi}{4}$ avec une vitesse nulle.

Supposons que le moment d'inertie en précession de l'ensemble {toupie, carter} soit de l'ordre de $j_p = 0.08$ kg.m^2 (légèrement supérieur à j_T). Le couple développé pour faire précessionner un actionneur gyroscopique est donné par :

$$c_p = j_p \cdot \ddot{\sigma} = 0.08 \times \frac{6\pi}{4}(2t-1)$$

Son maximum est atteint pour $t = 0$ et $t = 1$ seconde et vaut :

$$c_{p_{max}} = 0.377 \text{ N.m}$$

Or la vitesse de rotation $-\Omega\vec{u}$ du satellite induit un moment gyroscopique résistant autour des axes de précession de chacun des deux actionneurs gyroscopiques, donné par :

$$C_r = |-\Omega\vec{u} \times \vec{h}| = h \cdot \Omega \cdot \cos(\sigma)$$

Or

$$I \cdot \Omega = 2h \cdot \sin(\sigma)$$

d'où

$$C_r = \frac{h^2}{I}\sin(2\sigma)$$

qui atteint un maximum :

$$C_{r_{max}} = 25 \text{ N.m}$$

pour $t = 1$ s lorsque $\sigma = \frac{\pi}{4}$ rd. Ainsi la puissance totale maximale nécessaire pour faire précessionner un actionneur gyroscopique est :

$$P_{max} = p_{max} + C_{r_{max}} \cdot \Omega_{max} = 0.17 + 25 \times 1 = 25.17 \text{ W}$$

et l'énergie dépensée à cet effet vaut :

$$E_p = \int_0^1 P_{max}\, dt = 25.17 \text{ J}$$

On dépense donc légèrement plus d'énergie inutile ($2 \times 25.17 = 50.34$ J pour faire précessionner les toupies) que d'énergie utile (50 J pour faire tourner le satellite), et le rendement reste donc proche de 50%. Les gyrodynes présentent donc, au niveau énergétique, un avantage considérable sur les roues à inertie.

2.2.3 Grappes de gyrodynes

Afin de contrôler l'attitude du satellite selon les trois axes de roulis, tangage et lacet, on a recours à des grappes d'actionneurs gyroscopiques constituées d'un nombre $N \geq 3$ de gyrodynes, d'axes de précession de directions fixes $\vec{z}_i$ relativement au satellite (pour $i = 1$ à N). Notons $\vec{x}_i$ les axes de rotation propre des toupies qui tournent dans des plans fixes relativement au satellite, orthogonaux aux axes $\vec{z}_i$. Notons h le module du moment cinétique, supposé constant, de chaque toupie, et σ_i l'angle de rotation de l'axe $\vec{x}_i$ autour de l'axe $\vec{z}_i$, relativement à sa position initiale $\vec{x}_{i0}$.

Le moment cinétique résultant des toupies est donné par :

$$\vec{h} = h \sum_{i=1}^{N} \vec{x}_i$$

Le moment dynamique produit par la variation du moment cinétique des toupies s'écrit :

$$\vec{m} = \frac{d}{dt}_{|R_C} \vec{h} = \frac{d}{dt}_{|R_S} \vec{h} + \vec{\Omega} \times \vec{h}$$

où $\frac{d}{dt}_{|R_C}$ est la dérivation relativement au repère inertiel, $\frac{d}{dt}_{|R_S}$ la dérivation relativement au repère satellite et $\vec{\Omega}$ est la vitesse du satellite relativement au repère inertiel.

Le couple gyroscopique moteur $\vec{m}_g$ produit par les gyrodynes correspond au terme $\frac{d}{dt}_{|R_S} \vec{h}$:

$$\vec{m}_g = \frac{d}{dt}_{|R_S} \vec{h} = \frac{d}{dt}_{|R_S} \sum_{i=1}^{N} \vec{x}_i = \sum_{i=1}^{N} \frac{d\vec{x}_i}{d\sigma_i} = \sum_{i=1}^{N} \vec{y}_i \cdot \dot{\sigma}_i$$

car $\frac{d\vec{x}_i}{d\sigma_i} = \dot{\sigma}_i \vec{z}_i \times \vec{x}_i = \dot{\sigma}_i \vec{y}_i$

Notons $[y_i]$ les matrices colonnes des composantes des vecteurs $\vec{y}_i$ dans le repère satellite, et posons :

$$\begin{aligned} Y &= ([y_1][y_2]...[y_N]) \\ \dot{\sigma} &= \begin{pmatrix} \dot{\sigma}_1 \\ \dot{\sigma}_2 \\ \vdots \\ \dot{\sigma}_N \end{pmatrix} \end{aligned}$$

Si $[m_g]$ est la matrice colonne des composantes du vecteur $\vec{m_g}$ dans le repère satellite, alors :

$$[m_g] = Y\dot{\sigma}$$

Configuration singulière On dit qu'une grappe est dans une configuration singulière σ^s quand pour cette configuration, il existe une direction $\vec{u}$ telle que quel que soit $\dot{\sigma}$, le moment dynamique produit $\vec{m}_g$ est toujours orthogonal à $\vec{u}$. Autrement dit, on ne peut trouver de $\dot{\sigma}$ tel que le produit scalaire $\langle \vec{u} \cdot \vec{m}_g \rangle$ soit non nul. Dans une telle configuration, la grappe est dans l'incapacité de produire un couple dans la direction $\vec{u}$.

Or, pour que

$$\langle \vec{u} \cdot \vec{m}_g \rangle = [u]^T[m_g] = [u]^T Y\dot{\sigma}$$

soit nul pour tout $\dot{\sigma}$, il faut et il suffit que

$$[u]^T Y = 0$$

La grappe est en configuration singulière lorsque les N vecteurs $\vec{y}_i$ sont coplanaires, dans le plan orthogonal au vecteur $\vec{u}$.

Dans le cas d'une grappe composée de trois actionneurs gyroscopiques disposés en trièdre tri-rectangle, on ne peut utiliser que 16.9% de la capacité maximale en moment cinétique de la grappe pour être sûr de ne pas tomber sur une singularité. Afin d'augmenter la capacité de ce type d'actionneur, on utilise plus couramment des grappes redondantes (dont le nombre d'actionneurs gyroscopiques, N, est supérieur à 3), et on définit des stratégies d'évitement des singularités comme cela a été fait dans [Busseuil *et al.*, 1998].

Simplification Pour simplifier les calculs, dans la suite du chapitre nous considérons que les mouvements en attitude du satellite sont indépendants selon les trois axes de roulis, tangage et lacet. Cette indépendance est assurée par l'utilisation soit de roues à inertie, soit de trois paires d'actionneurs gyroscopiques contrôlant chacune un axe de rotation.

3 Nécessité d'un guidage bord en attitude

Que ce soit pour observer une zone au sol, détecter sa couverture nuageuse, recharger les batteries, télédécharger les données d'observation vers une station sol, effectuer une manœuvre orbitale, pointer le centre de la Terre, ou encore réaliser un rendez-vous en attitude, toutes les actions du satellite agile nécessitent que l'engin suive une trajectoire précise en attitude.

Dans le cas d'une prise de décision en ligne et à bord, un satellite agile d'observation doit être capable de générer lui-même les trajectoires en attitude nécessaires à la réalisation de ses différentes actions.

Le problème de décision décrit dans le chapitre 3 consiste, pour le satellite, à décider à chaque instant de la meilleure prochaine action à entreprendre, étant donnés l'état courant du satellite, de ses objectifs et l'information la plus récente sur la couverture nuageuse terrestre. Pour prendre une telle décision, le satellite doit connaître les préconditions (ou faisabilités) et les effets de chaque action possible. Dans la plupart des problèmes de planification, accéder aux préconditions et effets d'une action est simple et immédiat ; la difficulté du problème réside dans le choix de l'action parmi l'ensemble des actions possibles. Dans notre problème de décision, accéder à ces préconditions et effets est déjà en lui-même un problème difficile car il impose de raisonner sur les mouvements en attitude du satellite nécessaires pour débuter chaque action et pour la réaliser.

En effet, la production d'énergie par les panneaux solaires du satellite, au cours d'une action, dépend de l'orientation de ces panneaux par rapport aux rayons du Soleil ; elle dépend donc de la trajectoire en attitude suivie par le satellite au cours de cette action. La consommation d'énergie au cours d'une action dépend également de la trajectoire en attitude suivie au cours de cette action : elle sera d'autant plus grande que la durée et la vitesse des mouvements en attitude seront grands. Ces production et consommation d'énergie ont un impact immédiat sur les effets de chaque action. En outre, une action a ne sera possible à un instant t que si l'attitude du satellite et sa vitesse de rotation sont compatibles avec un début de l'action a à la date t (l'antenne de télédéchargement, par exemple, doit être contenue dans le cône de visibilité station dans le cas d'une action de télédéchargement). Ceci a un impact direct sur la faisabilité de chaque action.

En plus d'un besoin de génération et de suivi de trajectoire à bord du satellite, au moment de l'exécution d'une action, il apparaît donc également à bord, un besoin de prévisions nécessaires à la tâche de planification : prévisions des durées des actions de rendez-vous en attitude et prévisions des productions et des consommations d'énergie. Ces prévisions passent inévitablement par une prévision à bord des trajectoires en attitude à suivre au cours des activités du satellite. Dans la section suivante nous nous intéressons en particulier à l'estimation des trajectoires en attitude pour des rendez-vous à durée minimale.

4 Estimation des durées minimales de rendez-vous

Les trajectoires en attitude qui doivent être suivies par le satellite au cours des actions d'observation ou de manœuvre orbitale sont déterminées au sol, car elles font intervenir des calculs complexes (notamment dus à la courbure de la Terre dans le cas d'une observation). Au cours d'une action de rechargement des batteries, l'attitude du satellite reste constante[3], et dans le cas d'une action de détection de la couverture nuageuse (pointage du satellite 30 ° en avant), de pointage géocentrique ou de télédéchargement, les trajectoires en attitude sont relativement simples à obtenir à bord car il s'agit de garder le satellite pointé dans une certaine direction. Les actions qui demandent les calculs de trajectoire en attitude les plus difficiles à effectuer à bord sont principalement les actions de rendez-vous en attitude. Dans cette section, nous décrivons le problème d'estimation des trajectoires de rendez-vous en attitude à durée minimale que nous avons rencontré, puis l'approche que nous avons proposée pour le résoudre [Beaumet *et al.*, 2007a,b].

4.1 Description

Le sous-problème d'estimation de la durée minimale d'un rendez-vous en attitude peut être formulé de la façon suivante :
Étant donnés :

- un *état initial* caractérisé par l'attitude et la vitesse de rotation du satellite à la fin de l'action courante,
- un *état but* caractérisé par des contraintes sur la date, l'attitude et la vitesse de rotation du satellite au début de la prochaine action,
- des *contraintes sur les mouvements en attitude* comme les moments et couples maximaux des actionneurs,

le problème de décision consiste à déterminer s'il existe un rendez-vous en attitude permettant d'atteindre l'état but. S'il en existe un, le problème d'optimisation associé consiste à calculer sa durée minimale.

Les contraintes sur l'état but dépendent de la nature de la prochaine action à réaliser. Par exemple, si la prochaine action est un rechargement des batteries, l'état but doit être atteint avant que le satellite ne soit en éclipse, l'attitude visée est fixe (panneaux solaires dirigés vers le Soleil) et la vitesse de rotation du satellite est nulle (voir section 4.5). Si la prochaine action est un télédéchargement de données vers une station sol, l'état but doit être atteint avant la fin de la fenêtre de visibilité entre le satellite et la station, et l'attitude et la vitesse de rotation visées sont fonction de l'instant auquel l'état but sera atteint car le satellite se déplace constamment sur son orbite, et la Terre tourne sur elle-même. Ce problème peut être vu comme un problème de poursuite de cible mobile par un véhicule mobile. Il en est de même si la prochaine action est l'observation d'une zone au

[3]On suppose que la position du Soleil et du nœud ascendant de l'orbite du satellite sont constants au cours d'une demi-orbite éclairée (durée maximale d'une action de rechargement des batteries).

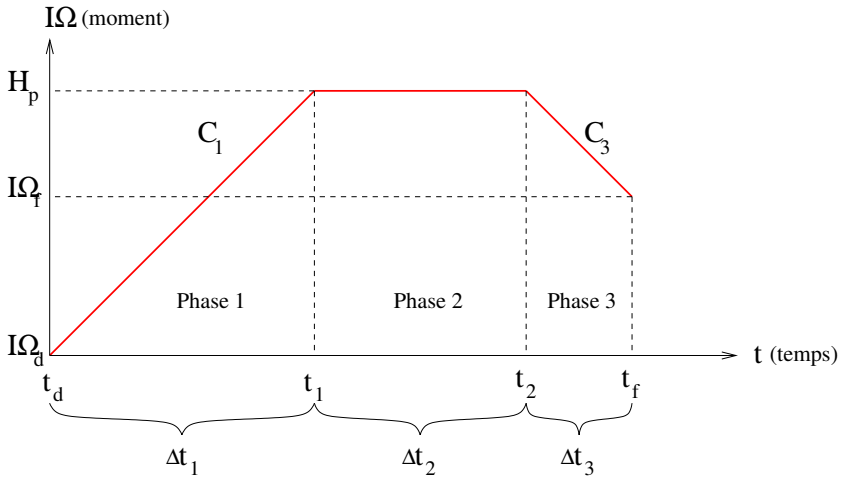


FIG. 4.7: Les trois phases du mouvement

sol a, avec des contraintes supplémentaires sur l'attitude et la vitesse de rotation que le satellite cherche à atteindre, pour imposer à la barrette de détecteurs de son instrument d'observation, de balayer a dans la bonne direction et à la bonne vitesse (voir section 4.6).

Dans le cas d'un satellite agile équipé d'une grappe de trois ou quatre actionneurs gyroscopiques, les mouvements selon les trois axes de roulis, tangage et lacet sont couplés. Pour simplifier le problème nous considérons ici que ces trois mouvements sont indépendants, c'est-à-dire que le satellite est soit équipé de trois roues à réactions, soit de trois paires d'actionneurs gyroscopiques commandant chacune la rotation de l'engin selon un axe. On impose de plus que chacun de ces mouvements soit décomposé en trois phases successives (cf figure 4.7) :

- une première phase à couple constant C_1,
- un palier à moment constant H_2,
- et une troisième phase à couple constant C_3.

Une telle stratégie peut être justifiée de la manière suivante : pour arriver le plus vite possible dans une certaine attitude, il est nécessaire d'accélérer jusqu'à une vitesse maximale (deux premières phases). Mais pour atteindre au même instant la vitesse de rotation visée, il peut être nécessaire de terminer le mouvement avec une phase de décélération (troisième phase). Pour se faire une idée du problème, le lecteur peut imaginer qu'il conduit un véhicule sur une route, et qu'il souhaite rattraper le plus vite possible et suivre, le véhicule qui est devant lui.

En plus des dates de début t_d de la première phase et de fin t_f de la dernière phase qui doivent évidemment être identiques sur les trois axes du mouvement, nous imposons également que les dates t_1 et t_2 de la phase de commutation soient identiques sur les trois axes, afin de limiter les perturbations dues aux accélérations et décélérations instantanées.

Le problème consiste alors à déterminer les trois dates de fin de phases t_1, t_2 et t_f, et, sur chaque axe, les couples C_1 et C_3 au cours des première et dernière phases, et le moment H_2 au cours du palier. Dans la suite du chapitre, nous modélisons et résolvons ce problème comme un problème de CSP continu.

4.2 Méthodes existantes

Des méthodes numériques existent pour résoudre ce type de problème. En particulier le CNES a développé la librairie MANIAC (MANœuvre Imposée en Attitude sous Contraintes) [Parraud *et al.*, 2006] qui contient un algorithme dédié aux calculs de rendez-vous en attitude pour plusieurs types de plates-formes satellites dont l'attitude est contrôlée par des roues à réaction (MYRIADE, PROTEUS, ...). Cet algorithme permet le calcul de la faisabilité des rendez-vous en attitude à durée fixée entre un pointage de référence et une prise de vue, ou entre deux prises de vues. Un algorithme itératif permet de résoudre le problème des rendez-vous à durée minimale. Dans le cadre du développement des futurs satellites Pléiades, dont l'attitude sera contrôlée grâce à des actionneurs gyroscopiques, le CNES a développé une librairie similaire nommée GOTlib (Guidage pour l'Observation de la Terre).

Cependant, ces méthodes présentent les inconvénients suivants :

- elles ont été développées pour être utilisées au sol, dans le cadre d'une construction régulière de plans d'activités pour le satellite ; elles consomment donc beaucoup de ressources et leurs temps d'exécution sont longs et incompatibles avec une utilisation en ligne ;
- elles reposent souvent sur la recherche des zéros d'une certaine fonction à l'aide de mécanismes de recherche locale de type Newton ou Lagrange ; elles n'offrent par conséquent aucune garantie d'optimalité sur la durée d'un rendez-vous en attitude ;
- elles n'utilisent pas de modèle explicite du problème à résoudre : état initial, état but et contraintes sur les mouvements en attitude ; ce modèle est noyé dans les algorithmes.

4.3 Approche proposée

Le raisonnement sur les intervalles a déjà été utilisé en robotique [Merlet, 2006]. Dans la même lignée, notre approche consiste à partir d'un modèle simplifié de la cinématique des différents corps qui entrent en jeu (Soleil, Terre, satellite) et à utiliser un solveur de CSP continu afin d'estimer le mieux possible la durée minimale d'un rendez-vous en attitude. Les principaux avantages a priori d'une telle méthode sont :

- l'utilisation d'un modèle explicite du problème, cohérente avec l'approche à base de modèles souhaitée dans le projet AGATA, et appliquée à tous les niveaux d'autonomie de l'engin (décision et suivi de situation) ;
- la possibilité d'utiliser ce même modèle pour répondre à diverses requêtes, simplement en changeant le critère d'optimisation (cf section 4.8) ;

– la possibilité d'obtenir non plus seulement un optimum local du problème, mais un optimum global, ou tout du moins un encadrement de l'optimum global.

4.4 Modèle cinématique

Dans cette section, nous présentons le modèle simplifié de la cinématique Terre - Soleil - satellite, que nous utilisons pour résoudre le problème d'estimation de la durée minimale d'un rendez-vous en attitude.

4.4.1 Mouvement de la Terre autour du Soleil

Nous avons modélisé le mouvement de la Terre autour du Soleil de différentes manières, prenant en compte ou non le retard du à la vitesse de propagation de la lumière, le fait que l'excentricité et l'obliquité de l'orbite terrestre ne sont pas constantes, et l'effet de l'attraction lunaire. Pour les échelles de temps considérées au cours de notre étude, l'utilisation du modèle le plus simplifié est suffisant : la trajectoire de la Terre autour du Soleil est donc supposée être une ellipse parfaite dont l'excentricité e et l'obliquité[4] ϵ au 1er janvier 2000 sont données par :

$$\begin{aligned} e &= 0.016709114 \\ \epsilon &= 0.40909261 \text{ rd} \end{aligned}$$

L'anomalie moyenne $M(t)$ de la Terre à l'instant t est donnée par :

$$M(t) = \frac{1}{\tau_{ano}}(t - t_{po})$$

où τ_{ano} est la période de temps qui sépare deux passages consécutifs de la Terre à son périhélie, appelée *année anomalistique* et estimée actuellement à $\tau_{ano} = 365.2596445$ jours, et t_{po} est une date de passage au périhélie. L'anomalie moyenne est ainsi relative au périhélie de l'année en cours.

On calcule alors directement l'anomalie vraie de la Terre, $\nu(t)$, à l'instant t, à partir de son anomalie moyenne $M(t)$ en utilisant la relation approchée suivante, valable pour de faibles excentricités e :

$$\nu(t) \simeq M(t) + C$$

où

$$C = (2e - \frac{1}{4}e^3)\sin(M) + (\frac{5}{4}e^2 - \frac{11}{24}e^4)\sin(2M) + \frac{13}{12}e^3 \cdot \sin(3M) + \frac{103}{96}e^4 \cdot \sin(4M)$$

[4]L'obliquité est l'inclinaison de l'écliptique moyen sur l'équateur moyen.

On en déduit la distance Terre - Soleil $\rho(t)$, à l'instant t, donnée par l'équation de l'ellipse :

$$\rho(t) = \frac{p}{1 + e \cdot \cos(\nu(t))} = \frac{a(1-e^2)}{1 + e \cdot \cos(\nu(t))}$$

p et a étant respectivement le paramètre et le demi-grand axe de l'ellipse. ($a = 1.0000002$ Unités Astronomiques[5]).

La longitude écliptique $\Lambda_\odot(t)$ du Soleil à l'instant t est donnée par :

$$\Lambda_\odot(t) = \Lambda_{\odot_0} + \frac{1}{\tau_{tro}} t + C$$

où $\Lambda_{\odot_0}$ est sa longitude écliptique à l'instant $t = 0$, soit $\Lambda_{\odot_0} = 280.4659^\circ$; et τ_{tro} est la période de temps qui sépare deux passages consécutifs de la Terre au point vernal, γ (qui précessionne par rapport aux étoiles), appelée *année tropique* et estimée actuellement à : $\tau_{tro} = 365.24218967$ jours solaires.

Il en résulte la position $\overrightarrow{T\odot}(t)$ du Soleil dans le repère inertiel $\mathcal{R}_C$ à l'instant t :

$$\overrightarrow{T\odot}(t) = \rho(t) \begin{bmatrix} \cos(\Lambda_\odot(t)) \\ \sin(\Lambda_\odot(t)) \cos(\epsilon) \\ \sin(\Lambda_\odot(t)) \sin(\epsilon) \end{bmatrix}^{R_C}$$

4.4.2 Mouvement d'une zone à la surface de la Terre

Une zone à observer est définie par un point M à la surface de la Terre et un cap (direction d'observation). La latitude lat_M du point M est fixe et l'ascension droite $as_M(t)$ du point M à un instant t est définie par :

$$as_M(t) = as_{M_d} + \Omega_T(t - t_d)$$

où as_{M_d} est l'ascension droite du point M à la date t_d et $\Omega_T = \frac{2\pi}{86400}$rd.s^{-1} la vitesse de rotation propre de la Terre. L'orientation du repère zone $\mathcal{R}_A$ est donnée par la matrice de passage du repère inertiel au repère zone :

$$M_{CA}(t) = R_\theta(-\frac{\pi}{2}) R_\psi(\pi) R_\psi(as_M(t)) R_\theta(lat_M) R_\phi(cap)$$

Ainsi la position $\overrightarrow{TM}(t)$ du point M dans $\mathcal{R}_\mathcal{C}$ à l'instant t est donnée par le vecteur :

$$\overrightarrow{TM}(t) = M_{CA}(t) \begin{bmatrix} 0 \\ 0 \\ R_T \end{bmatrix}^{R_A} = \begin{bmatrix} R_T c_{lat_M} c_{as_M(t)} \\ R_T c_{lat_M} s_{as_M(t)} \\ R_T s_{lat_M} \end{bmatrix}^{R_C}$$

[5] 1 Unité Astronomique = 149597870691 mètres

où $R_T = 6378.13$ km. Sa vitesse $\overrightarrow{V}_M(t)$ est donnée par :

$$\begin{aligned} \overrightarrow{V}_M(t) &= \overrightarrow{\Omega}_{CA} \wedge \overrightarrow{TM}(t) \\ &= \Omega_T \vec{z}_C \wedge \overrightarrow{TM}(t) \end{aligned}$$

4.4.3 Mouvement du satellite autour de la Terre

Mouvement de l'orbite du satellite autour de la Terre L'orbite circulaire héliosynchrone du satellite est définie par son rayon R_o et son inclinaison i. L'ascension droite $\Omega(t)$ de son noeud ascendant[6] à l'instant t est donnée par :

$$\Omega(t) = \Omega_d + \dot{\Omega}(t - t_d)$$

où Ω_d est l'ascension droite du noeud ascendant à la date t_d et $\dot{\Omega}$ sa vitesse de rotation constante autour de $\vec{z}_C$.

Mouvement du satellite sur son orbite La position du satellite sur son orbite à l'instant t est définie par son anomalie vraie[7] $\nu(t)$ donnée par :

$$\nu(t) = \nu_d + \dot{\nu}(t - t_d) = \nu_d + \frac{2\pi}{\tau}(t - t_d)$$

où ν_d est son anomalie vraie à la date t_d, $\dot{\nu}$ son mouvement moyen et τ la période de son orbite. L'orientation du repère orbital local $\mathcal{R}_L$ est donnée par la matrice de passage du repère inertiel au repère orbital local :

$$M_{CL}(t) = R_\phi(\Omega(t)) R_\psi(i) R_\phi(\frac{\pi}{2}) R_\psi(\frac{\pi}{2}) R_\theta(\nu(t))$$

Ainsi la position $\overrightarrow{TS}(t)$ du satellite dans R_C à l'instant t est donnée par le vecteur :

$$\overrightarrow{TS}(t) = M_{CL}(t) \begin{bmatrix} 0 \\ 0 \\ R_o \end{bmatrix}^{R_L}$$

Sa vitesse $\overrightarrow{V}_S(t)$ est donnée par :

$$\begin{aligned} \overrightarrow{V}_S(t) &= \overrightarrow{\Omega}_{CL} \wedge \overrightarrow{TS}(t) \\ &= (\dot{\Omega}\vec{z}_C + \dot{\nu}\vec{y}_L) \wedge \overrightarrow{TS}(t) \end{aligned}$$

[6] Point de l'orbite du satellite où celui-ci traverse l'écliptique depuis l'hémisphère Sud vers l'hémisphère Nord.

[7] Angle vu du centre T de la Terre entre la position du satellite et une position de référence sur son orbite.

4.4.4 Mouvement en attitude du satellite

Les mouvements en attitude du satellite sont contraints par sa matrice d'inertie I et les moments et couples maximaux délivrés par les actionneurs M_{max} et C_{max} :

$$I = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix} = \begin{pmatrix} 850 & 0 & 0 \\ 0 & 850 & 0 \\ 0 & 0 & 750 \end{pmatrix} \text{ m}^2.\text{kg}$$

$$\begin{bmatrix} H_{x_{max}} \\ H_{y_{max}} \\ H_{z_{max}} \end{bmatrix} = \begin{bmatrix} 45 \\ 45 \\ 20 \end{bmatrix}^{R_S} \text{ N.m} \qquad \begin{bmatrix} C_{x_{max}} \\ C_{y_{max}} \\ C_{z_{max}} \end{bmatrix} = \begin{bmatrix} 7 \\ 7 \\ 6 \end{bmatrix}^{R_S} \text{ N.m}$$

4.5 Rendez-vous à attitude fixée

Nous nous intéressons dans cette partie au cas simple du calcul d'un rendez-vous en attitude à durée minimale, vers une attitude fixe. Un tel rendez-vous est nécessaire lorsque le satellite décide de débuter au plus tôt une action de rechargement de ses batteries, en pointant ses panneaux solaires dans la direction du Soleil. À cause de la distance conséquente entre la Terre et le Soleil, et des propriétés de l'orbite héliosynchrone du satellite, l'attitude A_{Sol} du satellite requise par une action de pointage Soleil reste constante au cours d'une période éclairée de l'orbite du satellite (un peu plus d'une demi-révolution du satellite autour de la Terre ; voir figure 4.8).

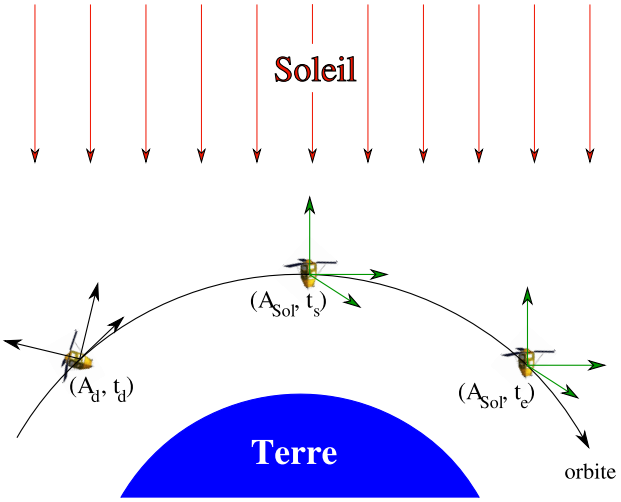


FIG. 4.8: Ralliement d'une action de pointage Soleil

Comme les mouvements en attitude du satellite selon les trois axes de roulis, tangage et lacet sont supposés indépendants, nous ne considérons par la suite que le mouvement en attitude selon un des trois axes.

Soit α_d (respectivement α_f) l'attitude[8] initiale (respectivement finale), avec $\Delta\alpha = \alpha_f - \alpha_d$. Soit Ω_d (respectivement Ω_f) la vitesse de rotation initiale (respectivement finale) du satellite. Soit t_d la date de début du rendez-vous en attitude, et t_1 (respectivement t_2 et t_f) la date de fin de la phase 1 (respectivement 2 et 3), avec $\Delta t_1 = t_1 - t_d$, $\Delta t_2 = t_2 - t_1$ et $\Delta t_3 = t_f - t_2$. Soit C_1 (respectivement C_3) le couple constant délivré par les actionneurs au cours de la phase 1 (respectivement 3) et H_2 le moment constant au cours de la phase 2 (cf. figure 4.7). Le moment cinétique du satellite, $I \cdot \Omega(t)$, à tout instant t est donné par les équations suivantes :

- au cours de la phase 1 : $I \cdot \Omega(t) = I \cdot \Omega_d + C_1(t - t_d)$
- au cours de la phase 2 : $I \cdot \Omega(t) = H_2$
- au cours de la phase 3 : $I \cdot \Omega(t) = I \cdot \Omega_f + C_3(t - t_f)$

La continuité du moment cinétique du satellite aux dates t_1 et t_2 impose que :

$$I \cdot \Omega_d + C_1(t_1 - t_d) = I \cdot \Omega_f + C_3(t_2 - t_f) = H_2$$

D'où l'on tire les équations 4.3 et 4.4 :

$$C_1 = \frac{H_2 - I \cdot \Omega_d}{\Delta t_1} \tag{4.3}$$

$$C_3 = \frac{I \cdot \Omega_f - H_2}{\Delta t_3} \tag{4.4}$$

On obtient alors la variation de l'angle de rotation $\Delta\alpha$ entre les dates t_d et t_f :

$$\begin{aligned} I \cdot \Delta\alpha &= \int_{t_d}^{t_f} I \cdot \Omega(t)\, dt \\ &= I \cdot \Omega_d \cdot \Delta t_1 + C_1 \cdot \frac{{\Delta t_1}^2}{2} + H_2 \cdot \Delta t_2 + I \cdot \Omega_f \cdot \Delta t_3 + C_3 \cdot \frac{{\Delta t_3}^2}{2} \end{aligned}$$

En remplaçant C_1 et C_3 par leurs expressions données par les équations 4.3 et 4.4, on obtient :

$$I \cdot \Delta\alpha = \frac{H_2 + I \cdot \Omega_d}{2}\Delta t_1 + H_2 \cdot \Delta t_2 + \frac{H_2 + I \cdot \Omega_f}{2}\Delta t_3 \tag{4.5}$$

De l'équation 4.5, on tire :

$$H_2 = \frac{2I \cdot \Delta\alpha - I \cdot \Omega_d \cdot \Delta t_1 - I \cdot \Omega_f \cdot \Delta t_3}{\Delta t_1 + 2\Delta t_2 + \Delta t_3} \tag{4.6}$$

[8]Selon un seul axe, cette attitude correspond à un angle.

Les contraintes $-H_{max} \leq H_2 \leq H_{max}$ s'écrivent :

$$cL_1 : (H_{max} - I \cdot \Omega_d)\Delta t_1 + 2H_{max} \cdot \Delta t_2 + (H_{max} - I \cdot \Omega_f)\Delta t_3 + 2I \cdot \Delta\alpha \geq 0$$

$$cL_2 : (H_{max} + I \cdot \Omega_d)\Delta t_1 + 2H_{max} \cdot \Delta t_2 + (H_{max} + I \cdot \Omega_f)\Delta t_3 - 2I \cdot \Delta\alpha \geq 0$$

En reportant la valeur de H_2 dans les équations 4.3 et 4.4, les contraintes $-C_{max} \leq C_1, C_3 \leq C_{max}$ s'écrivent :

$$-C_{max} \leq \frac{2I \cdot \Delta\alpha - I \cdot \Omega_d \cdot \Delta t_1 - I \cdot \Omega_f \cdot \Delta t_3}{\Delta t_1(\Delta t_1 + 2\Delta t_2 + \Delta t_3)} - \frac{I \cdot \Omega_d}{\Delta t_1} \leq C_{max}$$
$$-C_{max} \leq \frac{I \cdot \Omega_d \cdot \Delta t_1 + I \cdot \Omega_f \cdot \Delta t_3 - 2I \cdot \Delta\alpha}{\Delta t_3(\Delta t_1 + 2\Delta t_2 + \Delta t_3)} + \frac{I \cdot \Omega_d}{\Delta t_3} \leq C_{max}$$

soit :

$$cQ_1 : 2I \cdot \Delta\alpha - 2I \cdot \Omega_d \cdot \Delta t_1 - 2I \cdot \Omega_d \cdot \Delta t_2 - (I \cdot \Omega_d + I \cdot \Omega_f)\Delta t_3 + C_{max} \cdot \Delta t_1(\Delta t_1 + 2\Delta t_2 + \Delta t_3) \geq 0$$

$$cQ_2 : -2I \cdot \Delta\alpha + 2I \cdot \Omega_d \cdot \Delta t_1 + 2I \cdot \Omega_d \cdot \Delta t_2 + (I \cdot \Omega_d + I \cdot \Omega_f)\Delta t_3 + C_{max} \cdot \Delta t_1(\Delta t_1 + 2\Delta t_2 + \Delta t_3) \geq 0$$

$$cQ_3 : +2I \cdot \Delta\alpha - 2I \cdot \Omega_f \cdot \Delta t_2 - 2I \cdot \Omega_f \cdot \Delta t_3 - (I \cdot \Omega_d + I \cdot \Omega_f)\Delta t_1 + C_{max} \cdot \Delta t_3(\Delta t_1 + 2\Delta t_2 + \Delta t_3) \geq 0$$

$$cQ_4 : -2I \cdot \Delta\alpha + 2I \cdot \Omega_f \cdot \Delta t_2 + 2I \cdot \Omega_f \Delta t_3 + (I \cdot \Omega_d + I \cdot \Omega_f)\Delta t_1 + C_{max} \cdot \Delta t_3(\Delta t_1 + 2\Delta t_2 + \Delta t_3) \geq 0$$

Le problème consiste alors à minimiser $t_f = t_d + \Delta t_1 + \Delta t_2 + \Delta t_3$ sous les deux contraintes linéaires cL_1 et cL_2 et les quatre contraintes quadratiques cQ_1 à cQ_4.

4.6 Rendez-vous à attitude variable

En dehors d'une action de rechargement des batteries, l'attitude et la vitesse de rotation que le satellite souhaite atteindre n'est jamais fixe dans le repère inertiel. Elles sont variables et dépendent de la date t_f à laquelle le rendez-vous en attitude se termine. C'est le cas pour une action d'observation, un télédéchargement de données, une détection de la couverture nuageuse, un pointage géocentrique ou une manœuvre orbitale.

Nous considérons ici le cas le plus contraint, correspondant au ralliement d'une action d'observation d'une zone a à la surface de la Terre. La figure 4.9 indique comment l'attitude

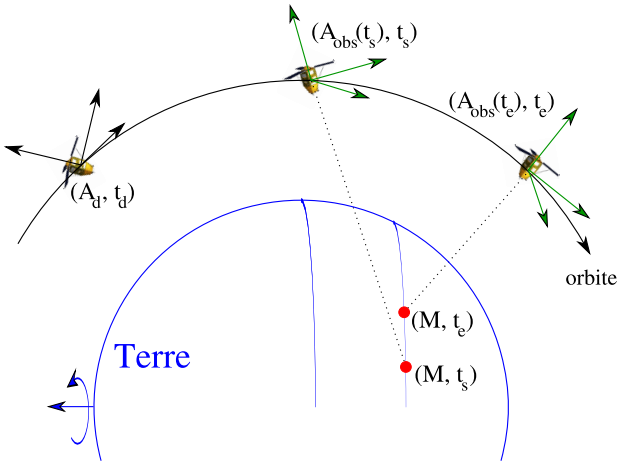


FIG. 4.9: Ralliement d'une action d'observation

de début d'observation visée dépend de la date de début de l'observation à l'intérieur de la fenêtre de visibilité $[t_s; t_e]$ de la zone a. Toutes les contraintes linéaires et quadratiques décrites dans la section précédente pour le cas d'un rendez-vous vers une attitude fixe restent valables. α_f et Ω_f ne sont plus constantes. Elles sont maintenant variables et doivent respecter les contraintes de début d'observation suivantes :

- l'instrument d'observation doit être dirigé vers le point de départ M de la zone a à observer :

$$\vec{z}_S = \frac{\overrightarrow{TS}(t_f) - \overrightarrow{TM}(t_f)}{||\overrightarrow{TS}(t_f) - \overrightarrow{TM}(t_f)||}$$

- la barrette de détecteurs composant l'instrument d'observation doit être orthogonale à la direction d'observation de la zone a :

$$\vec{x}_S = \vec{y}_A(t_f) \times \vec{z}_S$$

  Ainsi l'orientation du repère satellite $\mathcal{R}_S$ à la date t_f est donnée par la matrice de passage du repère inertiel $\mathcal{R}_C$ à $\mathcal{R}_S$:

$$M_{cs} = \begin{pmatrix} \vec{x}_S & \vec{z}_S \times \vec{x}_S & \vec{z}_S \end{pmatrix}$$

- la vitesse de la projection au sol de la barrette de détecteurs de l'instrument d'observation doit être égale à la somme de la vitesse de déplacement de la zone au sol et de la vitesse de balayage v_{bal} nécessaire pour observer la zone :

$$\vec{V}_S(t_f) + \vec{\Omega}_{CS}(t_f) \times (\overrightarrow{TM}(t_f) - \overrightarrow{TS}(t_f)) = \vec{V}_M(t_f) + v_{bal}\vec{x}_A(t_f)$$

où $\vec{\Omega}_{CS}(t_f)$ est le vecteur rotation du repère satellite $\mathcal{R}_S$ par rapport au repère inertiel $\mathcal{R}_C$ à la date t.

4.7 Modèle CSP

Le problème peut alors être converti en le problème de satisfaction de contraintes continu suivant [Granvilliers et Benhamou, 2006a].

Variables représentant le mouvement en attitude :

$t_f \in [t_s; t_e]$	$C_{1x}, C_{3x} \in [-C_{maxx}; C_{maxx}]$	$H_{2x} \in [-H_{maxx}; H_{maxx}]$
$t_1, t_2 \in [t_d; t_e]$	$C_{1y}, C_{3y} \in [-C_{maxy}; C_{maxy}]$	$H_{2y} \in [-H_{maxy}; H_{maxy}]$
	$C_{1z}, C_{3z} \in [-C_{maxz}; C_{maxz}]$	$H_{2z} \in [-H_{maxz}; H_{maxz}]$

Variables représentant l'état but à la date t_f :

$M_{CA} \in \mathcal{M}_3(\mathbb{R})$	$\vec{x}_A \in [-1,1]^3$	$\vec{y}_A \in [-1,1]^3$
$M_{CL} \in \mathcal{M}_3(\mathbb{R})$	$\vec{\Omega}_{CS} \in]-\infty; +\infty[^3$	$\vec{\Omega}_{CS}^{R_S} \in]-\infty; +\infty[^3$
$M_{CS} \in \mathcal{M}_3(\mathbb{R})$	$\vec{x}_S \in [-1,1]^3$	$\vec{z}_S \in [-1,1]^3$
$\Delta\phi \in [-\pi; \pi]$	$\Delta\theta \in [-\pi; \pi]$	$\Delta\psi \in [-\pi; \pi]$

Contraintes :

$t_d \leq t_1 \leq t_2 \leq t_f$

cL_1, cL_2, cQ_1, cQ_2, cQ_3, cQ_4 selon l'axe $\vec{x}_S$

cL_1, cL_2, cQ_1, cQ_2, cQ_3, cQ_4 selon l'axe $\vec{y}_S$

cL_1, cL_2, cQ_1, cQ_2, cQ_3, cQ_4 selon l'axe $\vec{z}_S$

$M_{CA} = M_{CA}(t_f) = R_\theta(-\frac{\pi}{2}) R_\phi(\pi) R_\phi(as_M(t_f)) R_\theta(lat_M) R_\psi(cap)$

$\vec{x}_A = M_{CA}\vec{x}_C \qquad \vec{y}_A = M_{CA}\vec{y}_C$

$M_{CL} = M_{CL}(t_f) = R_\psi(\Omega(t_f)) R_\phi(i) R_\psi(\frac{\pi}{2}) R_\phi(\frac{\pi}{2}) R_\theta(\nu(t_f))$

$\vec{z}_S = \frac{\overrightarrow{TS}(t_f) - \overrightarrow{TM}(t_f)}{||\overrightarrow{TS}(t_f) - \overrightarrow{TM}(t_f)||} \qquad \vec{x}_S = \vec{y}_A \times \vec{z}_S$

$M_{CS} = \begin{pmatrix} \vec{x}_S & \vec{z}_S \times \vec{x}_S & \vec{z}_S \end{pmatrix} \qquad \vec{\Omega}_{CS} = M_{CS}\vec{\Omega}_{CS}^{R_s}$

$\vec{V}_S(t_f) + \vec{\Omega}_{CS} \times (\overrightarrow{TM}(t_f) - \overrightarrow{TS}(t_f)) = \vec{V}_M(t_f) + v_{bal}\vec{x}_A$

$\Delta\phi = \arctan(\frac{(M_{CS})_{32}}{(M_{CS})_{33}}) \qquad \Delta\theta = -\arcsin((M_{CS})_{31}) \qquad \Delta\psi = \arctan(\frac{(M_{CS})_{21}}{(M_{CS})_{11}})$

4.8 Résolution avec l'outil RealPaver

Pour estimer la durée minimale d'un rendez-vous en attitude, nous avons utilisé le solveur de contraintes continu RealPaver [Granvilliers et Benhamou, 2006b]. Dans cette section nous présentons un exemple de calcul de la durée minimale d'un rendez-vous en attitude où le satellite est dans l'attitude de départ A_d (équation 4.7) à la date $t_d = 0$ et la prochaine action à entreprendre est l'observation d'un point M ($lat_M = 40.1\,^\circ$, $as_{M_d} = 1.0$ rd et $cap = 0.5$ rd) avec une vitesse de balayage nulle ($v_{bal} = 0$ m.s^{-1}) dans la fenêtre de visibilité $[t_s; t_e] = [0; 100]$ s. Le code source de cet exemple se trouve à l'annexe A.

L'orbite considérée est caractérisée par son rayon $R_o = 7204.8$ km, son inclinaison $i = 98.72\,^\circ$, la vitesse de rotation de son noeud ascendant $\dot{\Omega} = 1.99 \cdot 10^{-7}$ rd.s^{-1}, et l'ascension droite de son noeud ascendant $\Omega_d = -2.23$ rd à la date t_d. Le mouvement moyen du satellite est $\dot{\nu} = 1.03 \cdot 10^{-3}$ rd.s^{-1} et son anomalie vraie à la date t_d est $\nu_d = 2.3$ rd.

$$A_d = \{ \begin{bmatrix} \psi_d \\ \theta_d \\ \phi_d \end{bmatrix}, \begin{bmatrix} \dot{\psi}_d \\ \dot{\theta}_d \\ \dot{\phi}_d \end{bmatrix} \} = \{ \begin{bmatrix} 1.34 \\ -0.0682 \\ -0.254 \end{bmatrix} \text{ rd}, \begin{bmatrix} 0.0495 \\ 0.0133 \\ 0.0494 \end{bmatrix} \text{ rd.s}^{-1} \} \tag{4.7}$$

```
OUTER BOX: HULL of 1 boxes
  tf            in [16.75 , 16.77]
  delta_psi     in [-1.5 , -1.499]
  delta_theta   in [-0.1763 , -0.1761]
  delta_phi     in [1.144 , 1.146]
  C1x           in [-7 , -0.1952]
  C3x           in [-7 , +7]
  C1y           in [-7 , +7]
  C3y           in [-7 , +7]
  C1z           in [-6 , -1.016]
  C3z           in [-6 , +6]
  H2x           in [-45 , +41.59]
  H2y           in [-45 , +45]
  H2z           in [-20 , +20]
  t1            in [2.84 , 16.77]
  t2            in [2.84 , 16.77]
  [...]

  precision: 90, elapsed time: 240 ms
```

FIG. 4.10: Borne inférieure de la durée minimale

```
OUTER BOX: HULL of 1 boxes
  tf            in [49.02 , 49.04]
  delta_psi     in [-1.552 , -1.551]
  delta_theta   in [-0.3708 , -0.3704]
  delta_phi     in [1.207 , 1.209]
  C1x           = -7 **point**
  C3x           = 7 **point**
  C1y           in [-1.697 , -1.688]
  C3y           in [0.5936 , 0.6169]
  C1z           in [-1.398 , -1.382]
  C3z           in [-3.478 , -3.413]
  H2x           in [-45 , -44.91]
  H2y           in [-9.745 , -9.662]
  H2z           in [19.68 , 19.84]
  t1            in [12.43 , 12.45]
  t2            in [43.27 , 43.31]
  [...]

  precision: 59.6, elapsed time: 600 ms
```

FIG. 4.11: Borne supérieure de la durée minimale

La figure 4.10 représente le calcul d'une *borne inférieure* de la durée minimale du rendez-vous en attitude ($\Delta T = t_f - t_d \approx 16.75$ s) calculée en 240 ms. Ce résultat nous permet de savoir uniquement qu'il n'existe pas de rendez-vous en attitude de durée plus courte que 16.75 secondes, et ne fournit aucune information sur la trajectoire en attitude optimale.

Comme le montre la figure 4.11, il est possible d'obtenir une *borne supérieure* de la durée minimale de rendez-vous en attitude en supprimant les deux degrés de liberté restants sur la trajectoire. Une idée cohérente serait de fixer la valeur des couples C_1 et C_3 aux valeurs maximales d'accélération et de décélération, selon l'axe sur lequel le débattement angulaire est le plus grand. En fixant $C_{1x} = -C_{x_{max}}$ and $C_{3x} = C_{x_{max}}$ nous obtenons une borne supérieure de la durée minimale du rendez-vous en attitude ($t_f = 49.04$ s) ainsi que les paramètres définissant la trajectoire en attitude à suivre (C_1, C_3 et H_2). Cependant, les temps de calcul (quelques dixièmes de secondes) sont trop longs et incompatibles avec un raisonnement en ligne qui demande un très grand nombre de calculs de ce genre. Afin de remédier à ce problème il resterait à explorer d'autres possibilités de fixer les degrés de liberté restants, afin de minimiser les temps de calcul nécessaires.

```
OUTER BOX: HULL of 1 boxes
  cos_a-1       in [1.036 , 1.038]
  tf            in [107.4 , 107.5]
  delta_psi     in [1.363 , 1.365]
  delta_theta   in [-0.7951 , -0.7948]
  delta_phi     in [1.349 , 1.351]
  C1x           in [-7 , +1.02]
  C3x           in [-7 , +7]
  C1y           in [-7 , +7]
  C3y           in [-7 , +7]
  C1z           in [-6 , -0.1586]
  C3z           in [-6 , +6]
  t1            in [2.84 , 107.5]
  t2            in [2.84 , 107.5]
  [...]

  precision: 105, elapsed time: 4,760 ms
```

FIG. 4.12: Optimisation de l'angle de prise de vue

Grâce au modèle explicite que nous avons défini, il est en outre possible d'optimiser d'autres critères que la durée minimale d'un rendez-vous en attitude. Si le satellite a beaucoup d'observations à réaliser en peu de temps, il semble raisonnable de chercher à minimiser les durées des rendez-vous en attitude. Mais si les observations sont peu nombreuses, il peut être plus intéressant de chercher à optimiser leurs qualités. Une manière

d'y arriver consiste à minimiser leurs angles de prise de vue[9] : plus l'angle de prise de vue est petit, meilleure sera la qualité de l'image réalisée. La figure 4.12 montre le résultat obtenu lors du calcul d'une borne inférieure du nouveau critère `cos_a-1` $= \frac{1}{<\vec{z_S},\vec{z_L}>}$ obtenu en 4.760 ms, avec $t_f - t_d \simeq 107.4$ s.

4.9 Synthèse des résultats

Les points importants à retenir de ce travail sont les suivants :

- un des principaux avantages d'utiliser un outil de programmation par contraintes est d'imposer l'écriture d'un modèle explicite du problème à résoudre, qui peut servir de modèle de référence pour l'élaboration des algorithmes de résolution ; de plus, ce même modèle peut être utilisé tel quel, avec différents critères d'optimisation ;
- contrairement aux méthodes de recherche locale, le raisonnement sur les intervalles permet le calcul d'un encadrement de l'optimum ; dans le cas étudié précédemment, une simple borne inférieure de la durée minimale d'un rendez-vous en attitude permet d'éliminer immédiatement des actions candidates impossibles ;
- cependant, la séparation récursive des domaines des variables utilisée par l'outil de programmation par contraintes ne permet pas de calculer des solutions optimales en temps raisonnable ; en pratique, des choix heuristiques seront nécessaires afin d'obtenir rapidement des solutions de qualité raisonnable ;
- dans le modèle utilisé nous avons utilisé le formalisme des matrices de rotation, faisant appel à un grand nombre de fonctions trigonométriques ; afin de limiter ces fonctions coûteuses en temps de calcul, une voie restante à explorer est l'utilisation des quaternions dans la modélisation.

5 Fonctions de calcul de trajectoires en attitude effectivement implémentées

Les temps de calcul nécessaires au solveur de contraintes continu utilisé, ne sont actuellement pas compatibles avec un calcul en ligne des trajectoires en attitude optimales. C'est pourquoi, dans cette section, nous décrivons la méthode itérative d'estimation de ces trajectoires en attitude que nous avons effectivement implémentée dans le logiciel de vol du satellite (voir chapitre 5).

Étant donné un satellite S ayant une attitude A_d et une vitesse de rotation Ω_d à une date t_d, on cherche ici à déterminer le rendez-vous en attitude de durée minimale, nécessaire pour rallier une action a entre les dates t_s et t_e. La figure 4.9 illustre le cas où l'action a est une action d'observation.

[9] L'angle de prise de vue est l'angle entre la direction $\overrightarrow{SM}$ et le nadir.

5.1 Calcul de la durée d'un rendez-vous à attitude fixe

Dans cette section, on cherche à calculer le temps minimal de basculement pour faire passer le satellite d'une attitude A et une vitesse de rotation Ω_A à une attitude B et une vitesse de rotation Ω_B.

On note $\mathcal{B}_A$ et $\mathcal{B}_B$ les bases correspondant au repère satellite lorsque celui-ci se trouve respectivement dans l'attitude A et dans l'attitude B. A et B sont repérées par deux quaternions d'attitude Q_{OA} et Q_{OB} relativement à une attitude de référence O. Les vitesses Ω_A et Ω_B sont connues dans le repère satellite, c'est-à-dire en composantes dans les bases $\mathcal{B}_A$ et $\mathcal{B}_B$: $\Omega_A^{\mathcal{B}_A}$ et $\Omega_B^{\mathcal{B}_B}$.

On a alors :

$$\begin{aligned} \dot{Q}_A^{\mathcal{B}_A} &= \frac{1}{2}\Omega_A^{\mathcal{B}_A} \cdot Q_{OA} \\ \dot{Q}_B^{\mathcal{B}_B} &= \frac{1}{2}\Omega_B^{\mathcal{B}_B} \cdot Q_{OB} \end{aligned}$$

On note $Q_{AB} = \bar{Q}_{OA} \cdot Q_{OB}$ le quaternion de rotation de $\mathcal{B}_A$ à $\mathcal{B}_B$, exprimé indifféremment dans $\mathcal{B}_A$ ou $\mathcal{B}_B$. Notons α_{AB} et $\vec{u}_{AB}$ l'angle et le vecteur unitaire de l'axe de cette rotation :

$$Q_{AB} = (\cos(\frac{\alpha_{AB}}{2}), \sin(\frac{\alpha_{AB}}{2})\vec{u}_{AB})$$

Considérons la base intermédiaire $\mathcal{B}_I$ à mi-chemin entre $\mathcal{B}_A$ et $\mathcal{B}_B$. On a :

$$\begin{aligned} Q_{AI} &= Q_{IB} = (\cos(\frac{\alpha_{AB}}{4}), \sin(\frac{\alpha_{AB}}{4})\vec{u}_{AB}) \\ Q_{IA} &= \bar{Q}_{AI} \end{aligned}$$

Q_{IA} et Q_{IB} repèrent les attitudes A et B relativement à la base intermédiaire $\mathcal{B}_I$. En composantes dans cette base, on a :

$$\begin{aligned} \dot{Q}_A^{\mathcal{B}_I} &= Q_{IA} \cdot \dot{Q}_A^{\mathcal{B}_A} \cdot Q_{AI} = \frac{1}{2} Q_{IA} \cdot \Omega_A^{\mathcal{B}_A} \cdot Q_{OA} \cdot Q_{AI} = \frac{1}{2} Q_{IA} \cdot \Omega_A^{\mathcal{B}_A} \cdot Q_{OI} \\ \dot{Q}_B^{\mathcal{B}_I} &= Q_{IB} \cdot \dot{Q}_B^{\mathcal{B}_B} \cdot Q_{BI} = \frac{1}{2} Q_{IB} \cdot \Omega_B^{\mathcal{B}_B} \cdot Q_{OB} \cdot Q_{BI} = \frac{1}{2} Q_{IB} \cdot \Omega_B^{\mathcal{B}_B} \cdot Q_{OI} \end{aligned}$$

Les temps de basculement minimaux sont alors calculés dans la base intermédiaire $\mathcal{B}_I$, de manière indépendante sur les trois composantes vectorielles du quaternion courant Q_{IM} évoluant de Q_{IA} à Q_{IB}, à l'aide d'une loi de commande de type "bang bang" [LaValle, 2006] implémentée dans un algorithme, `Tmin` [Llibre, 2007]. Le temps minimal global de basculement nécessaire au satellite pour passer d'une attitude A et une vitesse de rotation Ω_A à une attitude B et une vitesse de rotation Ω_B, correspond au plus grand de ces trois temps.

Les accélérations et vitesses maximales sur les trois composantes vectorielles de $\dot{Q}_M^{\mathcal{B}_I}$ sont respectivement prises égales à la moitié des accélérations et vitesses de rotation

maximales sur les axes de roulis, tangage et lacet du satellite. Cette approximation est relativement bonne car la base $\mathcal{B}_I$ définit l'attitude moyenne du satellite au cours du basculement. Ainsi, si Q_{OM} est proche de Q_{OI} alors :

$$Q_{IM}^{\mathcal{B}_I} \simeq \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Donc

$$\begin{aligned} \dot{Q}_M^{\mathcal{B}_I} &= \frac{1}{2}\Omega \cdot Q_{IM} \\ &\simeq \frac{1}{2}\Omega \\ \dot{Q}_{max}^{\mathcal{B}_I} &\simeq \frac{1}{2}\Omega_{max} \end{aligned}$$

Il en est de même pour :

$$\begin{aligned} \ddot{Q}_M^{\mathcal{B}_I} &= \frac{1}{2}(\dot{\Omega} \cdot Q_{IM} + \Omega \cdot \dot{Q}_M^{\mathcal{B}_I}) \\ &= \frac{1}{2}(\dot{\Omega} \cdot Q_{IM} + \Omega \cdot (\frac{1}{2}\Omega \cdot Q_{IM})) \\ &\simeq \frac{1}{2}\dot{\Omega} \cdot Q_{IM} \\ &\simeq \frac{1}{2}\dot{\Omega} \\ \ddot{Q}_{max}^{\mathcal{B}_I} &\simeq \frac{1}{2}\dot{\Omega}_{max} \end{aligned}$$

Si en plus d'une limitation par axe, la vitesse de basculement du satellite est limitée en module (du type $||\Omega|| < \Omega_{max}$), les valeurs de chaque composante de la partie vectorielle de $\dot{Q}_M^{\mathcal{B}_I}$ sont limitées aux valeurs absolues des composantes correspondantes de $\Omega_{max}\vec{u}_{AB}$.

5.2 Interpolation de la durée d'un rendez-vous à attitude variable

On cherche maintenant à généraliser le calcul des durées des rendez-vous aux rendez-vous à attitude variable en interpolant les durées calculées pour des rendez-vous à attitude fixe. Les dates t_s et t_e sont les instants de début au plus tôt et au plus tard de la prochaine action à réaliser, a. Les attitudes et vitesses de rotation du satellite correspondantes sont notées (A_{f_s}, Ω_{f_s}) et (A_{f_e}, Ω_{f_e}).

On représente un rendez-vous en attitude débutant à la date t_d avec l'attitude A_d et la vitesse de rotation Ω_d, par un triplet (t, A, Ω) où t, A et Ω sont respectivement sa

date, son attitude et sa vitesse de rotation finales. Le triplet de rendez-vous $(t_k, A_{f_k}, \Omega_{f_k})$ est calculé par l'algorithme `Tbascul` (cf. algorithme 4.1) décrit ci-après, faisant appel de façon itérative à l'algorithme `Tmin`, de calcul de la durée minimale d'un rendez-vous à attitude fixe, décrit dans la section 5.1. L'algorithme `Tbascul` procède par dichotomie sur l'intervalle $[t_s; t_e]$ afin de déterminer le temps de basculement minimum, sous l'hypothèse que si un rendez-vous en attitude est possible en une durée d, alors il est possible pour toute durée d' supérieure à d.

Ces travaux ont donné lieu à une communication nationale [Beaumet *et al.*, 2007b] aux Journées Françaises de Programmation par Contraintes (JFPC'07) et une communication internationale [Beaumet *et al.*, 2007a] à la conférence CP'07 (International Conference on Principles and Practice of Constraint Programming).

Algorithm 4.1 Durée minimale d'un rendez-vous à attitude variable

1: *% fonction Tbascul(t_d, A_d, Ω_d, t_s, t_e, ϵ) : durée minimale*
2: Calcul des attitude et vitesse nécessaires pour débuter l'action envisagée à la date t_s : A_{f_s} et Ω_{f_s}
3: $\tau_1 = \texttt{Tmin}(A_d, \Omega_d, A_{f_s}, \Omega_{f_s})$
4: **si** $\tau_1 < t_s$ **alors**
5: Le rendez-vous est réalisé avec le triplet $(t_s, A_{f_s}, \Omega_{f_s})$
6: **retourner** t_s
7: **sinon**
8: $\tau_2 = \texttt{Tmin}(A_d, \Omega_d, A_{f_e}, \Omega_{f_e})$
9: **si** $\tau_2 > t_e$ **alors**
10: Le rendez-vous est impossible dans la fenêtre $[t_s; t_e]$
11: **retourner** ÉCHEC
12: **sinon**
13: $t_1 \leftarrow t_s$
14: $t_2 \leftarrow t_e$
15: Interpolation d'une nouvelle date t_k de fin de rendez-vous : $t_k = \frac{\tau_1 \cdot t_2 - \tau_2 \cdot t_1}{(t_2 - t_1) - (\tau_2 - \tau_1)}$
16: Calcul des attitude et vitesse nécessaires pour débuter l'action envisagée à la date t_k : A_{f_k} et Ω_{f_k}
17: $\tau_k = \texttt{Tmin}(A_d, \Omega_d, A_{f_k}, \Omega_{f_k})$
18: **si** $\tau_k > t_k$ **alors**
19: $t_1 \leftarrow t_k$
20: $\tau_1 \leftarrow \tau_k$
21: On recommence à la ligne 15
22: **sinon**
23: **si** $\tau_k > t_k - \epsilon$ **alors**
24: La solution est trouvée à ϵ près
25: **retourner** τ_k
26: **sinon**
27: $t_2 \leftarrow t_k$
28: $\tau_2 \leftarrow \tau_k$
29: On recommence à la ligne 15
30: **finsi**
31: **finsi**
32: **finsi**
33: **finsi**

Chapitre 5

Gestion de la mission : architecture de contrôle autonome

Pour le contrôle autonome d'un satellite agile d'observation de la Terre, nous avons adapté, étendu et amélioré l'architecture générique réactive - délibérative, développée dans le cadre du projet AGATA [Beaumet *et al.*, 2008a,b] et présentée à la section 1.3.5 du chapitre 2.

1 Implémentation avec `Java` et `Esterel`

Nous avons utilisé le langage synchrone `Esterel` [Berry et Gonthier, 1992] pour implémenter les tâches réactives, le langage `Java` pour les tâches délibératives et le mécanisme de *task* offert par `Esterel` pour connecter tâches réactives et délibératives.

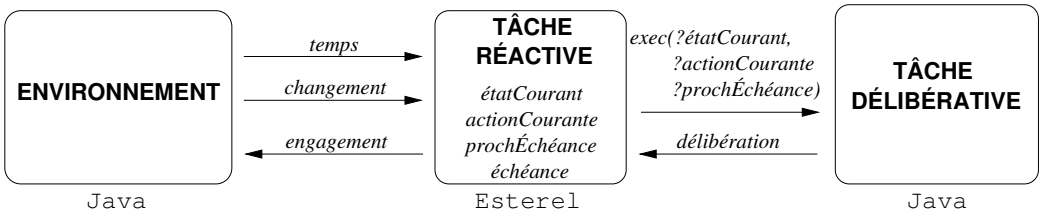


FIG. 5.1: Schéma fonctionnel des interactions entre l'environnement, une tâche réactive, et une tâche délibérative

La figure 5.1 représente les principaux évènements qui sont échangés entre l'environnement, une tâche réactive et une tâche délibérative (noms au dessus des flèches) et entre les modules de la tâche réactive (noms dans la boîte associée à la tâche réactive).

La tâche réactive reçoit de la part de l'environnement, un évènement *temps* qui l'informe du temps qui passe, et un évènement *changement* à chaque fois

qu'un changement se produit dans l'état de l'environnement. Elle envoie à l'environnement un évènement d'*engagement* informant l'environnement qu'une action vient d'être engagée à la suite d'une prise de décision. La tâche réactive peut également lancer l'exécution d'une tâche délibérative en l'informant sur l'état courant de l'environnement, l'action courante et la prochaine date de prise de décision (*exec(?étatCourant, ?actionCourante, ?prochÉchéance)*), et reçoit les *délibérations* de la tâche délibérative, à chaque fois qu'une meilleure solution est disponible. Les évènements échangés à l'intérieur de la tâche réactive sont *étatCourant* (informant de la mise à jour de l'état courant de l'environnement), *actionCourante* (informant de la mise à jour de l'action courante exécutée par le satellite), *prochÉchéance* (informant de la date de la prochaine prise de décision) et *échéance* (informant de l'arrivée d'une échéance à son terme).

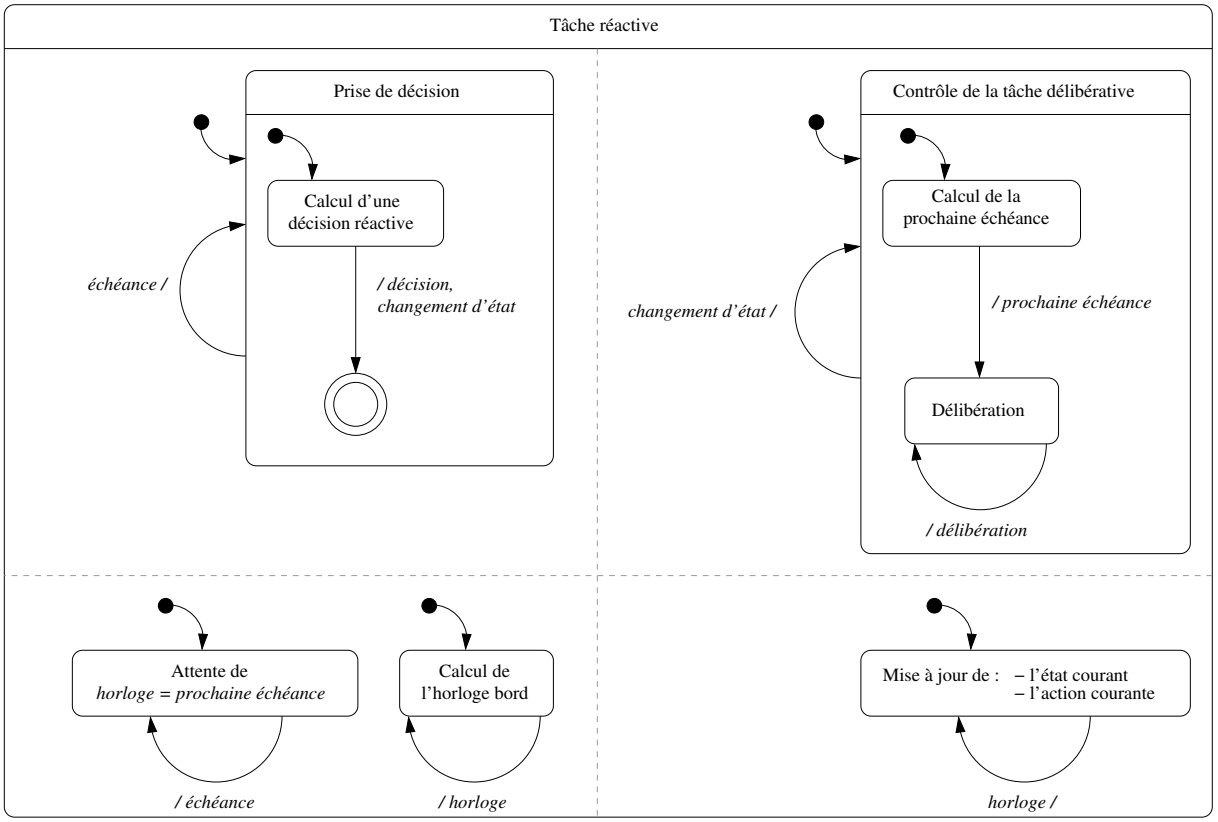


FIG. 5.2: Comportement simplifié de la tâche décisionnelle réactive

2 Tâche réactive

La tâche réactive joue le rôle d'interface entre l'environnement et la tâche délibérative :
– elle reçoit les stimuli en provenance de l'environnement ;

- elle calcule la date de la prochaine échéance (correspondant à la date de la prochaine prise de décision) ;
- elle contrôle la tâche délibérative (l'active, l'interrompt, l'abandonne, la relance) ;
- à chaque échéance, elle vérifie la cohérence de la délibération vis-à-vis de l'état courant du système, prend une décision et engage l'action correspondante ;
- et elle prend des décisions réactives lorsqu'aucune délibération n'est disponible au moment d'une prise de décision.

2.1 Structure de la tâche réactive

Le comportement de la tâche décisionnelle réactive est décrit sur la figure 5.2 sous la forme d'un diagramme SyncChart [Zaffalon, 2005]. Quatre mécanismes principaux s'exécutent en parallèle : la gestion du temps à bord du satellite, le suivi de l'état, la prise de décision et le contrôle de la tâche délibérative.

2.1.1 Gestion du temps

Une horloge bord permet de définir la date courante à bord du satellite. À chaque "tic" de l'horloge, le signal *horloge* est émis avec la valeur correspondant au nombre de secondes écoulées depuis une date référence. Le programme `Esterel` suivant décrit exactement ce comportement.

```
%%%=== Horloge interne
t0 := initialTime();
loop
   t := getTime(t0);
   emit horloge(t);
   pause;
end loop
```

`getTime(double t)` : temps écoulé depuis la date t.

Un module supplémentaire permet de comparer à chaque instant la date courante à bord et la date de la prochaine échéance. À chaque fois qu'une échéance arrive à son terme, le signal *échéance* est émis.

```
%%%=== Émission du signal d'échéance
loop
   if estEchue(?echeanceProchaine, ?horloge) then
      emit echeance;
   end if;
   pause;
end loop
```

2.1.2 Suivi de l'état

Un mécanisme de suivi de l'état permet de mettre à jour, à chaque instant :
- l'état courant du système, estimé à bord au travers des capteurs du satellite (c'est en partie sur cet état bord que se basent les mécanismes de décision et de planification) ;
- et l'action courante exécutée par le satellite (cette donnée permet notamment au satellite de calculer la prochaine échéance, c'est-à-dire la date de fin de l'action courante).

2.1.3 Prise de décision

À chaque fois que le signal *échéance* est émis, un mécanisme supposé a priori instantané (ce qui devra être vérifié une fois le logiciel de vol implémenté) se charge de la prise d'une décision en tenant compte des éventuelles délibérations émises auparavant, et émet les signaux *décision* et *changement d'état* afin de signaler qu'une décision vient d'être prise et que l'état du satellite vient de changer suite à l'engagement d'une action. Le programme `Esterel` suivant décrit ce comportement.

```
%%%=== Prise de décision à chaque échéance
every echeance do
   call calculDecisionReactive(d)(?etat_interne, ?deliberation,
                                  ?horloge, ?plan_courant);
   emit decisionReactive(d);
   emit engagement(d);
end every
```

`calculDecisionReactive(Decision d)(Etat e, Decision d, double t, Plan p)` : calcule la décision à prendre en fonction de l'état courant, la dernière délibération, la date courante, et le plan courant suivi par le satellite.

À chaque prise de décision, la tâche réactive vérifie la cohérence entre l'action décidée et l'état courant du système. Elle vérifie notamment que l'engagement de l'action ne remette pas en cause la survie du satellite (le niveau des batteries ne doit par exemple pas descendre en dessous du seuil critique), et que l'engagement de cette action dans l'état courant a toujours un sens (la date limite de réalisation d'une requête d'observation ne doit pas avoir expiré au moment de l'engagement de l'action d'observation correspondante par exemple).

2.1.4 Contrôle de la tâche délibérative

Enfin, à chaque changement d'état consécutif soit à une prise de décision, soit à un changement de l'état de l'environnement, une nouvelle échéance est calculée et la tâche délibérative est relancée. Celle-ci, lorsqu'elle se termine normalement, émet le signal *délibération* transmettant ainsi à la tâche réactive la meilleure prochaine décision

calculée jusqu'alors, et est immédiatement relancée. Elle peut également être avortée par la présence du signal *changement d'état*, auquel cas, une nouvelle échéance est calculée et la tâche délibérative est relancée. Les deux programmes `Esterel` suivants décrivent respectivement le mécanisme de calcul de la prochaine échéance suite à une prise de décision et le mécanisme de contrôle de la tâche délibérative par la tâche réactive.

```
%%%==== Traitement à chaque prise de décision
loop
   present preDecisionReactive then
      call amputePlanPremiereAction(plan)(?preDecisionReactive);
      emit changementEtat;
      t := majEcheanceProchaine(?preDecisionReactive);
      emit echeanceProchaine(t);
   end present;
   pause;
end loop
```

`amputePlanPremiereAction(Plan p)(Decision d)` : récupère le plan courant suivi par le satellite et l'ampute de sa première action, qui vient d'être engagée (voir la section 3.2 pour plus de détails).
`preDecisionReactive` indique que le signal `decisionReactive` était présent à l'instant précédent.
`majEcheanceProchaine(Decision d)` : calcule la prochaine échéance correspondant à la date de fin de l'action décidée.

```
%%%==== Contrôle de la tâche délibérative
every changementEtat do
   emit deliberation(DECISION_VIDE);
   if estEffective(?echeanceProchaine) then
      c := true;
      call computeDecisionState(etat_decision)(?etat_interne,
                                               ?action_courante);
      trap Optimum in
         loop
            exec deliberatif(d)(etat_decision, ?plan_courant, c)
            return retourDeliberation;
            emit deliberation(d);
            if(estOptimale(d)) then
               exit Optimum;
            end if;
            c := false
         end loop
      end trap
   end if
end every
```

`estEffective(double e)` : renvoie vrai si la prochaine échéance `e` est effective, c'est-à-dire si elle n'est pas encore atteinte.
`computeDecisionState(Etat eD)(Etat eI, Action a)` : calcule l'état dans lequel se trouvera le système à la fin de l'action courante.
`deliberatif(Decision d)(Etat eD, Plan p, boolean b)` : lance une nouvelle tâche délibérative visant à calculer la meilleure action à entreprendre dans l'état de décision.

2.1.5 Interface avec le simulateur du monde réel

Un cinquième mécanisme joue le rôle d'interface entre le logiciel de vol et le simulateur de la réalité : il gère l'évolution des états réel (simulé) et interne (estimé à bord) en fonction du temps qui passe et des évènements qui se produisent (prises de décisions, arrivées de nouvelles requêtes, générations de manœuvres orbitales, échecs d'observations, détections de la couverture nuageuse).

```
%%%=== Interface avec le simulateur du monde réel
  loop
     e := etat_reel(?horloge);
     call computeInnerState(e_interne)(e);
     present preDecisionReactive then
        call majEtatReelDecision(e)(?preDecisionReactive);
        call majEtatDecision(e_interne)(?preDecisionReactive);
     end present;

     present nouvelleRequete then
        call ajoutNouvelleRequete(e_interne)(?nouvelleRequete);
        emit changementEtat;
     end present;

     present nouvelleManoeuvre then
        call ajoutNouvelleManoeuvre(e_interne)(?nouvelleMan);
        emit changementEtat;
     end present;

     present echecObservation then
        call resetObservation(e_interne)(?echecObservation);
        emit changementEtat;
     end present;

     present detectionNuage then
        call majCouvertureNuageuse(e_interne)(?detectionNuage);
        emit changementEtat;
     end present;
```

```
    emit etat_interne(e_interne);

    a := action_courante();
    emit action_courante(a);

    pause;
  end loop
```

`computeInnerState(Etat eI, Etat eR)` : calcule l'état interne à partir de l'état réel envoyé par le simulateur (simule le rôle des capteurs).
À chaque prise de décision, les états réel et interne sont mis à jour grâce aux fonctions `majEtatReelDecision(Etat eR)` et `majEtatDecision(Etat eI)`.
À chaque arrivée d'une nouvelle requête d'observation, la fonction `ajoutNouvelleRequete (Etat eI)(Requete r)` ajoute cette requête à l'état interne du satellite.
À chaque génération d'une manœuvre orbitale par le contrôle d'attitude et d'orbite du satellite, la fonction `ajoutNouvelleManoeuvre(Etat eI)(Manoeuvre m)` ajoute cette manœuvre à l'état interne du satellite.
À chaque échec de réalisation d'une observation, l'état interne est mis à jour par la fonction `resetObservation(Etat eI)(Observation o)`.
À chaque arrivée d'informations sur la couverture nuageuse, l'état interne est enrichi grâce à la fonction `majCouvertureNuageuse(Etat eI)(DetectionNuage d)`.

Pour plus de détails sur l'implémentation de la partie réactive du logiciel de vol, on pourra se reporter à son code source à l'annexe B.

2.2 Règles de décision

Afin d'être capable de prendre des décisions réactives (c'est-à-dire en un temps suffisamment court pour qu'elles puissent être considérées comme instantanées au vu de la dynamique du système), nous avons mis en place quelques règles de décision permettant au logiciel de vol de prendre des décisions même en cas d'absence de délibération.

2.2.1 Définitions préalables

État de décision L'état de décision est l'état complet du système à la date à laquelle la décision doit être prise. Cet état décrit l'état du satellite (niveaux de ressources, attitude, etc.) et l'état de ses objectifs à cette date.

Horizon de décision L'horizon de décision est l'horizon temporel à partir de la date de l'état de décision, sur lequel on recense les actions envisageables.

Action envisageable Une action est envisageable à partir d'un état `e` et sur un certain horizon `h`, s'il existe un rendez-vous en attitude sur `h`, amenant le satellite de l'état `e` à un état `e'` dans lequel l'action peut débuter.

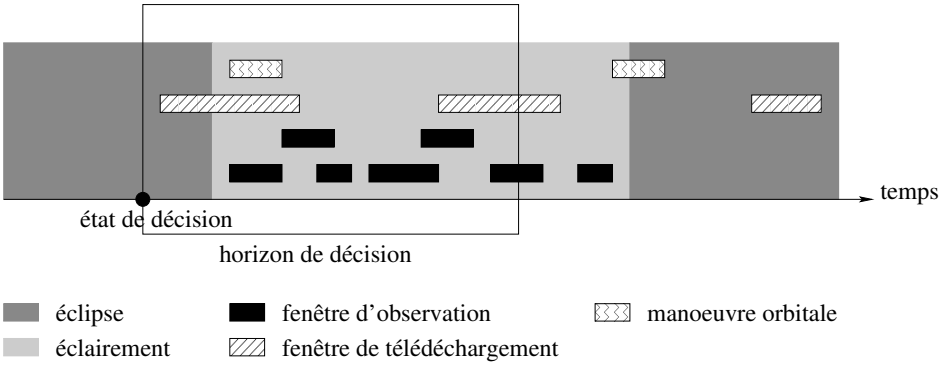


FIG. 5.3: Définitions : état, horizon de décision et action envisageable

2.2.2 Règle de décision 1 : choix de l'action la plus prioritaire

Principe général Soit `e` un état de décision et `h` un horizon de décision. On choisit d'engager dans l'état `e`, l'action la plus prioritaire, en parcourant l'ensemble des actions envisageables sur `h` par ordre de priorités décroissantes. Le tableau 5.1 indique les priorités associées aux différents types d'actions réalisables par le satellite.

action	priorité
manœuvre orbitale	6 (max)
télédéchargement	5
détection de la couverture nuageuse	4
observation	3
rechargement des batteries	2
pointage géocentrique	1 (min)

TAB. 5.1: Priorités des différents types d'actions

Choix d'une action de manœuvre orbitale On s'intéresse donc tout d'abord au type d'action le plus prioritaire : les actions de manœuvre orbitale. On détermine l'ensemble $\mathscr{L}_{\text{man}}$ des actions de manœuvre orbitale envisageables à partir de l'état `e` et sur l'horizon `h`.

Pour cela on énumère toutes les actions de manœuvre orbitale dont la date de début se situe entre la date de l'état de décision `e` et la fin de l'horizon `h`. Puis on ne garde que les actions pour lesquelles il existe un rendez-vous en attitude amenant le satellite de son attitude dans l'état `e` à l'attitude de début de la manœuvre.

Si $\mathscr{L}_{\text{man}}$ n'est pas vide, on choisit la manœuvre orbitale dont la date de début est la plus proche de l'état de décision (c'est un paramètre discriminant car il ne peut pas y avoir plus d'une manœuvre orbitale au même instant).

Algorithm 5.1 DR1 : Choix d'une action de manœuvre orbitale

```
% fonction manoeuvre(e, h) : manoeuvre
L ← ∅
pour man dans listeManoeuvre(e, h) faire
  t ← e.date + dureeMiniRdVMan(e.attitude, man)
  si t ≤ man.dateDebut alors
    L ← L ∪ {man}
  finsi
fin pour
retourner plusTot(L)
```

Choix d'une action de télédéchargement Si aucune manœuvre orbitale n'est possible, on détermine alors l'ensemble $\mathscr{L}_{\text{tel}}$ des actions de télédéchargement envisageables à partir de l'état `e` et sur l'horizon `h`.

Algorithm 5.2 DR1 : Choix d'une action de télédéchargement

```
% fonction teledechargement(e, h) : teledechargement
L ← ∅
Lobs ← ensemble des observations réalisées mais non télédéchargées
pour visibilite dans visibiliteCentreMission(e.date, h) faire
  t ← e.date + dureeMiniRdVTeledech(e.attitude, visibilite)
  si t < e.date + h alors
    teledech = teledech(visibilite.centreMission, listeObsPrioriteMax(Lobs))
    L ← L ∪ {teledech}
  finsi
fin pour
L_prioriteMax ← prioriteMax(L)
L_prioriteMaxPlusTot ← plusTot(L_prioriteMax)
retourner alphabetique(L_prioriteMaxPlusTot)
```

Pour cela on énumère l'ensemble des visibilités des centres de mission à partir de l'état `e` et sur l'horizon `h`. Pour chacune de ces visibilités, on construit une action de

télédéchargement vers le centre de mission correspondant, et on ne garde que les actions pour lesquelles il existe un rendez-vous en attitude amenant le satellite de son attitude dans l'état `e` à l'attitude de début du télédéchargement. Pour chacun de ces télédéchargements, on construit la liste des observations élémentaires à télédécharger en choisissant en priorité les observations de plus forte priorité (la priorité d'une requête d'observation est définie par un entier compris entre 1 et 3, 3 étant la priorité maximale) puis les observations dont la date de réalisation est la plus ancienne (à priorités et dates de réalisation égales, on choisit arbitrairement une observation élémentaire), jusqu'à remplir au maximum le créneau de visibilité du centre de mission.

Si l'ensemble $\mathscr{L}_{\text{tel}}$ n'est pas vide, on choisit l'action de télédéchargement qui contient l'observation élémentaire de plus grande priorité, puis celle dont la date de début est la plus proche de l'état de décision. Si ces deux paramètres ne sont pas discriminants, on choisit arbitrairement l'action de télédéchargement vers le centre de mission dont le nom est le premier dans l'ordre alphabétique.

Choix d'une action de détection Si aucun télédéchargement n'est possible, on détermine la première action de détection de la couverture nuageuse envisageable à partir de l'état `e` et sur l'horizon `h`.

Algorithm 5.3 DR1 : Choix d'une action de détection

% fonction detection(e, h) : detection
$t \leftarrow$ max(e.date + dureeMiniRdVDetection(e.attitude), dateDerniereDetection + Δ)
pour $d = t$ **à** e.date + h, **pas** = Δ **faire**
  si observationsNonDetectees(d) **alors**
    retourner detection(d)
  finsi
fin pour

Pour cela on calcule la date de fin au plus tôt, t, d'un rendez-vous en attitude amenant le satellite de son attitude dans l'état de décision `e` à l'attitude de début de détection. On évalue la durée de survol, par le satellite, d'une zone au sol dont la couverture nuageuse est détectée au cours d'une action de détection de la façon suivante :

$$\Delta = \frac{\texttt{altitudeSatellite}}{\cos(\texttt{angleTangageDetection} + \texttt{angleDemiLongueurDetection})} \times \frac{1}{\texttt{vitesseSolSatellite}}$$

On note t_d = max(t, date de la dernière détection + Δ). À partir de cette date t_d, on crée une action de détection toutes les Δ secondes et on choisit la première action de détection qui détecte la couverture nuageuse au dessus d'un ensemble de zones dont la

couverture nuageuse d'aucune n'a été récemment[1] détectée. Ceci interdit au satellite de détecter plusieurs fois de suite la couverture nuageuse au dessus d'une même zone.

Choix d'une action d'observation Si aucune action de détection n'est possible, on détermine l'ensemble $\mathscr{L}_o$ des actions d'observation envisageables à partir de l'état `e` et sur l'horizon `h`.

Pour cela on énumère l'ensemble des observations élémentaires visibles sur l'horizon `h` et non encore réalisées. Pour chacune de ces observations élémentaires, on construit une action d'observation et on ne garde que les actions pour lesquelles il existe un rendez-vous en attitude amenant le satellite de son attitude dans l'état `e` à une attitude de début de l'observation.

Si l'ensemble $\mathscr{L}_o$ n'est pas vide, on choisit l'observation élémentaire de plus grande priorité (priorité de la requête d'observation), puis l'observation élémentaire dont la date de début est la plus proche de l'état de décision. Si ces deux paramètres ne sont toujours pas discriminants, on choisit arbitrairement l'observation de plus petit index.

Algorithm 5.4 DR1 : Choix d'une action d'observation

```
% fonction observation(e, h) : observation
L ← ∅
pour obs dans listeObservation(e,h) faire
  t ← e.date + dureeMiniRdVObservation(e.attitude, obs)
  si estVisible(obs, t, e.date + h) alors
    si NON(estRealisee(obs)) alors
      L ← L ∪ {obs}
    finsi
  finsi
fin pour
L_prioriteMax ← maxPriorite(L)
L_prioriteMaxPlusTot ← plusTot(L_prioriteMax)
retourner minIndex(L_prioriteMaxPlusTot)
```

Choix d'une action de rechargement des batteries Si aucune action d'observation n'est possible, on détermine la prochaine action de rechargement des batteries envisageable à partir de l'état de décision `e` et sur l'horizon `h`.

Pour cela on calcule la date de fin au plus tôt, t, d'un rendez-vous en attitude amenant le satellite de son attitude dans l'état de décision `e` à l'attitude de début de pointage Soleil.

- Si à cette date t, le satellite est éclairé, alors on choisit une action de rechargement des batteries débutant à t et se terminant

[1]Une information de détection est récente si elle est arrivée au cours de la dernière révolution du satellite (voir la définition de la fraîcheur de l'information de détection à la section 3.1).

  - à la fin de l'horizon `h` si celui-ci est atteint avant le début de la prochaine éclipse ;
  - au début de la prochaine éclipse sinon.
- Si à cette date le satellite est en éclipse, alors
  - s'il existe une fenêtre d'éclairement du satellite sur l'intervalle de temps compris entre la date t et la fin de l'horizon h, on choisit une action de rechargement des batteries sur cette fenêtre d'éclairement ;
  - sinon, aucun rechargement des batteries n'est envisageable.

Algorithm 5.5 DR1 : Choix d'une action de rechargement des batteries

```
% fonction rechargementBatteries(e, h) : pointageSoleil
t ← e.date + dureeMiniRdVPointageSoleil(e.attitude)
si satelliteEclaire(t) alors
  prochaineEclipse ← prochaineEclipse(t)
  retourner pointageSoleil(t, MIN(prochaineEclipse, h))
sinon
  prochainEclairement ← prochainEclairement(t)
  si prochainEclairement < h alors
    prochaineEclipse ← prochaineEclipse(prohainEclairement)
    retourner pointageSoleil(prochainEclairement, MIN(prochaineEclipse, h))
  finsi
finsi
```

Choix d'une action de pointage géocentrique Si aucune action de rechargement des batteries n'est possible, on calcule la date de fin au plus tôt, t, d'un rendez-vous en attitude amenant le satellite de l'état de décision `e` à l'attitude de début de pointage géocentrique, et on choisit l'action de pointage géocentrique débutant à t et se terminant à la fin de l'horizon `h`.

Algorithm 5.6 DR1 : Choix d'une action de pointage géocentrique

```
% fonction pointageGeo(e, h) : pointageGeo
t ← e.date + dureeMiniRdVPointageGeo(e)
retourner pointageGeo(t, h)
```

Algorithme principal L'algorithme 5.7 représente le comportement général de la règle de décision 1.

Algorithm 5.7 Règle de décision 1

```
% fonction DR1(e, h) : action[]
si manoeuvre(e,h) ≠ ∅ alors
  actionPrincipale = manoeuvre(e,h)
sinon si teledechargement(e,h) ≠ ∅ alors
  actionPrincipale = teledechargement(e,h)
sinon si detection(e,h) ≠ ∅ alors
  actionPrincipale = detection(e,h)
sinon si observation(e,h) ≠ ∅ alors
  actionPrincipale = observation(e,h)
sinon si rechargementBatteries(e,h) ≠ ∅ alors
  actionPrincipale = rechargementBatterie(e,h)
sinon si pointageGeo(e,h) ≠ ∅ alors
  actionPrincipale = pointageGeo(e,h)
finsi
si estPossible(actionPrincipale, e) alors
  retourner [actionPrincipale]
sinon
  rdvAttitude = actionPrincipale.minimumRdVAttitudeNecessaire(e)
  retourner [rdvAttitude, actionPrincipale]
finsi
```

2.2.3 Règle de décision 2 : choix de l'action la plus proche de l'instant de décision

Principe général Soit `e` un état de décision et `h` un horizon de décision. Parmi les actions de manœuvre orbitale, de détection, d'observation, et de télédéchargement envisageables sur `h`, on choisit l'action dont la date de début au plut tôt est la plus proche de l'état de décision. S'il n'y en a pas, en période d'éclairement on choisit une action de rechargement des batteries, et en période d'éclipse on choisit une action de pointage géocentrique. À dates de début au plus tôt égales, on choisit l'action la plus prioritaire en respectant la hiérarchie suivante :

action	priorité
manœuvre orbitale	6 (max)
télédéchargement	5
détection de la couverture nuageuse	4
observation	3 (min)

TAB. 5.2: Priorités des différents types d'actions

À dates de début au plus tôt égales, et à priorités égales (même type d'action),
- pour une action d'observation, on choisit la requête d'observation la plus prioritaire. Si ce paramètre n'est pas discriminant, on choisit arbitrairement l'observation de plus petit index ;
- pour une action de télédéchargement, on choisit celle qui permet le télédéchargement de l'observation de plus grande priorité. Si ce paramètre n'est pas discriminant, on choisit arbitrairement le télédéchargement vers le centre de mission dont le nom est le premier dans l'ordre alphabétique.

Détermination de l'action de manœuvre orbitale débutant au plus tôt On détermine l'ensemble $\mathscr{L}_{\text{man}}$ des actions de manœuvre orbitale envisageables à partir de l'état `e` et sur l'horizon `h`.

Pour cela on énumère toutes les actions de manœuvre orbitale dont la date de début se situe entre la date de l'état de décision `e` et la fin de l'horizon `h`. Puis on ne garde que les actions pour lesquelles il existe un rendez-vous en attitude amenant le satellite de son attitude dans l'état `e` à l'attitude de début de la manœuvre.

Si l'ensemble $\mathscr{L}_{\text{man}}$ n'est pas vide, on choisit la manœuvre orbitale dont la date de début est la plus proche de l'état de décision (c'est un paramètre discriminant car deux manœuvres orbitales ne peuvent pas débuter au même instant).

Algorithm 5.8 DR2 : Détermination de l'action de manœuvre orbitale

```
% fonction manoeuvre(e, h) : manoeuvre
L ← ∅
pour man dans listeManoeuvre(e, h) faire
  t ← e.date + dureeMiniRdVMan(e.attitude, man)
  si t ≤ man.dateDebut alors
    L ← L ∪ {man}
  finsi
fin pour
retourner plusTot(L)
```

Détermination de l'action de télédéchargement débutant au plus tôt On détermine l'ensemble $\mathscr{L}_{\text{tel}}$ des actions de télédéchargement envisageables à partir de l'état `e` et sur l'horizon `h`.

Pour cela on énumère l'ensemble des visibilités des centres de mission à partir de l'état `e` et sur l'horizon `h`. Pour chacune de ces visibilités, on construit une action de télédéchargement vers le centre de mission correspondant, et on ne garde que les actions pour lesquelles il existe un rendez-vous en attitude amenant le satellite de son attitude dans l'état `e` à l'attitude de début du télédéchargement. Pour chacun de ces télédéchargements,

on construit la liste des observations élémentaires à télédécharger en choisissant en priorité les requêtes d'observation de plus forte priorité puis les observations dont la date de réalisation est la plus ancienne (à priorités et dates de réalisation égales, on choisit arbitrairement une observation élémentaire), jusqu'à remplir au maximum le créneau de visibilité.

Si l'ensemble $\mathscr{L}_{\text{tel}}$ n'est pas vide, on choisit le télédéchargement dont la date de début au plus tôt est la plus proche de l'état de décision. À dates de début au plus tôt égales, on choisit le télédéchargement qui contient l'observation élémentaire de plus grande priorité, puis arbitrairement le télédéchargement vers le centre de mission dont le nom est le premier dans l'ordre alphabétique.

Algorithm 5.9 DR2 : Détermination de l'action de télédéchargement

% fonction teledechargement(e, h) : teledechargement
$L \leftarrow \emptyset$
Lobs ← ensemble des observations réalisées mais non télédéchargées
pour *visibilite* **dans** visibiliteCentreMission(e.date, h) **faire**
  $t \leftarrow$ e.date + dureeMiniRdVTeledech(e.attitude, visibilite)
  si $t <$ e.date + h **alors**
    teledech = teledech(visibilite.centreMission, listeObsPrioriteMax(*Lobs*))
    $L \leftarrow L \cup \{teledech\}$
  finsi
fin pour
$L_{plusTot} \leftarrow$ plusTot(L)
$L_{plusTotPrioriteMax} \leftarrow$ prioriteMax($L_{plusTot}$)
retourner alphabetique($L_{plusTotPrioriteMax}$)

Détermination de l'action de détection débutant au plus tôt On détermine la première action de détection envisageable à partir de l'état `e` et sur l'horizon `h`.

Pour cela on calcule la date de fin au plus tôt, t, d'un rendez-vous en attitude amenant le satellite de son attitude dans l'état de décision `e` à l'attitude de début de détection. On évalue la durée de survol, par le satellite, d'une zone au sol dont la couverture nuageuse est détectée au cours d'une action de détection de la façon suivante :

$$\Delta = \frac{\texttt{altitudeSatellite}}{\cos(\texttt{angleTangageDetection} + \texttt{angleDemiLongueurDetection})} \times \frac{1}{\texttt{vitesseSolSatellite}}$$

On note $t_d = \max(t, \text{date de la dernière détection} + \Delta)$. À partir de cette date t_d, on crée une action de détection toutes les Δ secondes et on choisit la première action de détection qui détecte la couverture nuageuse au dessus d'un ensemble de zones dont la couverture nuageuse d'aucune n'a été récemment détectée.

Algorithm 5.10 DR2 : Détermination de l'action de détection

% fonction detection(e, h) : detection
$t \leftarrow$ max(e.date + dureeMiniRdVDetection(e.attitude), dateDerniereDetection + Δ)
pour $d = t$ **à** e.date + h, **pas** = Δ **faire**
  si observationsNonDetectees(d) **alors**
    retourner detection(d)
  finsi
fin pour

Détermination de l'action d'observation débutant au plus tôt On détermine l'ensemble $\mathscr{L}_o$ des actions d'observation envisageables à partir de l'état `e` et sur l'horizon `h`.

Pour cela on énumère l'ensemble des observations élémentaires visibles sur l'horizon `h` et non encore réalisées. Pour chacune de ces observations élémentaires, on construit une action d'observation et on ne garde que les actions pour lesquelles il existe un rendez-vous en attitude amenant le satellite de son attitude dans l'état `e` à une attitude de début de l'observation.

Si l'ensemble $\mathscr{L}_o$ n'est pas vide, on choisit l'observation élémentaire dont la date de début au plus tôt est la plus proche de l'état de décision. À dates de début au plus tôt égales, on choisit l'observation élémentaire de plus grande priorité (priorité de la requête d'observation), puis on choisit arbitrairement l'observation de plus petit index.

Algorithm 5.11 DR2 : Détermination de l'action d'observation

% fonction observation(e, h) : observation
$L \leftarrow \emptyset$
pour *obs* **dans** listeObservation(e,h) **faire**
  $t \leftarrow$ e.date + dureeMiniRdVObservation(e.attitude, *obs*)
  si estVisible(*obs*, t, e.date + h) **alors**
    si NON(estRealisee(*obs*)) **alors**
      $L \leftarrow L \cup \{obs\}$
    finsi
  finsi
fin pour
$L_{plusTot} \leftarrow$ plusTot(L)
$L_{plusTotPrioriteMax} \leftarrow$ prioriteMax($L_{plusTot}$)
retourner minIndex($L_{plusTotPrioriteMax}$)

Détermination de l'action de rechargement des batteries débutant au plus tôt Si aucune action de manœuvre orbitale, télédéchargement, détection ou observation n'est

possible sur l'horizon de décision, on détermine la prochaine action de rechargement des batteries envisageable à partir de l'état de décision `e` et sur l'horizon `h`.

Pour cela on calcule la date de fin au plus tôt, t, d'un rendez-vous en attitude amenant le satellite de son attitude dans l'état de décision `e` à l'attitude de début de pointage Soleil.

- Si à cette date le satellite est éclairé, alors on choisit une action de rechargement des batteries débutant à t et se terminant
  - à la fin de l'horizon `h` si celui-ci est atteint avant le début de la prochaine éclipse ;
  - au début de la prochaine éclipse sinon.
- Si à cette date le satellite est en éclipse, alors
  - s'il existe une fenêtre d'éclairement du satellite sur l'intervalle de temps compris entre la date t et la fin de l'horizon h, on choisit une action de rechargement des batteries sur cette fenêtre d'éclairement ;
  - sinon, aucun rechargement des batteries n'est envisageable.

Algorithm 5.12 DR2 : Détermination de l'action de rechargement des batteries

```
% fonction rechargementBatteries(e, h) : pointageSoleil
t ← e.date + dureeMiniRdVPointageSoleil(e.attitude)
si satelliteEclaire(t) alors
  prochaineEclipse ← prochaineEclipse(t)
  retourner pointageSoleil(t, MIN(prochaineEclipse, h))
sinon
  prochainEclairement ← prochainEclairement(t)
  si prochainEclairement < h alors
    prochaineEclipse ← prochaineEclipse(prohainEclairement)
    retourner pointageSoleil(prochainEclairement, MIN(prochaineEclipse, h))
  finsi
finsi
```

Détermination de l'action de pointage géocentrique débutant au plus tôt Si aucune action de rechargement des batteries n'est possible, on calcule la date de fin au plus tôt, t, d'un rendez-vous en attitude amenant le satellite de son attitude dans l'état de décision `e` à l'attitude de début de pointage géocentrique, et on choisit l'action de pointage géocentrique débutant à t et se terminant à la fin de l'horizon `h`.

Algorithm 5.13 DR2 : Détermination de l'action de pointage géocentrique

```
% fonction pointageGeo(e, h) : pointageGeo
t ← e.date + dureeMiniRdVPointageGeo(e)
retourner pointageGeo(t, h)
```

Algorithme principal L'algorithme 5.14 représente le comportement général de la règle de décision 2.

Algorithm 5.14 Règle de décision 2

```
% fonction DR2(e, h) : action[]
si plusTot(manoeuvre(e,h), teledechargement(e,h), detection(e,h),
observation(e,h)) = manoeuvre(e,h) alors
  actionPrincipale = manoeuvre(e,h)
sinon si plusTot(manoeuvre(e,h), teledechargement(e,h), detection(e,h),
observation(e,h)) = teledechargement(e,h) alors
  actionPrincipale = teledechargement(e,h)
sinon si plusTot(manoeuvre(e,h), teledechargement(e,h), detection(e,h),
observation(e,h)) = detection(e,h) alors
  actionPrincipale = detection(e,h)
sinon si plusTot(manoeuvre(e,h), teledechargement(e,h), detection(e,h),
observation(e,h)) = observation(e,h) alors
  actionPrincipale = observation(e,h)
sinon
  si rechargementBatteries(e,h) ≠ ∅ alors
    actionPrincipale = rechargementBatteries(e,h)
  sinon
    actionPrincipale = pointageGeo(e,h)
  finsi
finsi
si estPossible(actionPrincipale, e) alors
  retourner [actionPrincipale]
sinon
  rdvAttitude = actionPrincipale.minimumRdVAttitudeNecessaire(e)
  retourner [rdvAttitude, actionPrincipale]
finsi
```

2.2.4 Autres règles de décision

La règle de décision 1 (DR1) favorise la réalisation des actions les plus prioritaires. Si l'horizon de décision est grand, son principal inconvénient est qu'elle privilégie une action lointaine (en terme de temps écoulé à partir de l'instant de décision) de forte priorité à une action proche mais de priorité plus faible. Elle limite donc le nombre d'actions réalisées.

À l'opposé, la règle de décision 2 (DR2) favorise les actions proches de l'instant de décision. Elle privilégie ainsi une action de faible priorité pouvant débuter très rapidement, au risque d'empêcher la réalisation d'une action plus prioritaire débutant un peu plus tard.

Pour palier à ces deux inconvénients, nous avons proposé deux règles de décision supplémentaires, mixant les avantages des deux premières règles de décision :

- Règle de décision 3 (DR3) : elle est basée sur DR1 (choix de l'action a_1 la plus prioritaire) et tente en plus, d'insérer des actions de priorité inférieure débutant avant a_1 et n'empêchant pas son exécution ;
- Règle de décision 4 (DR4) : basée sur DR2 (choix de l'action a_1 la plus proche de l'instant de décision), cette règle de décision vérifie en plus, qu'aucune action plus prioritaire que a_1 n'est empêchée par la suite.

Les performances de ces différentes règles de décision dans les contextes en ligne et hors ligne, ainsi que le caractère "instantané" qu'elles doivent vérifier seront étudiés dans le chapitre 7.

3 Tâche délibérative

La tâche délibérative du logiciel de vol est chargée d'optimiser les décisions prises par le satellite en fonction du temps de raisonnement dont il dispose. Elle doit, par conséquent, avoir un comportement anytime [Zilberstein, 1996]. La tâche délibérative :

- est contrôlée par la tâche réactive ;
- est tenue informée par la tâche réactive, des données pertinentes sur le problème à résoudre (estimation du prochain état de décision, prochaine échéance, état des objectifs) ;
- tente de déterminer la meilleure prochaine action à entreprendre en optimisant successivement un plan sur un horizon limité en avant de la date de décision ;
- et rend des délibérations dès qu'elles sont disponibles et qu'elles améliorent strictement les précédentes.

3.1 Algorithme de planification

3.1.1 Principe général

Comme nous l'avons vu dans la section 1.4 du chapitre 3, nous avons choisi d'implémenter dans cette tâche délibérative, un algorithme glouton stochastique itéré qui consiste à réaliser une séquence de recherches gloutonnes stochastiques [Cicirello et Smith, 2005]. Chaque recherche gloutonne construit elle-même une séquence de décisions (et donc un plan) sur un horizon fini de planification. Dans le contexte d'une mission d'observation de la Terre, cet horizon peut s'étendre de la date de décision à la prochaine date de sortie d'éclipse afin de vérifier le bilan énergétique du satellite (car aucun rechargement des batteries n'est possible en période d'éclipse). Comme le montre la figure 5.4, chaque décision est prise en suivant une certaine heuristique qui est bruitée afin d'éviter aux différentes recherches gloutonnes de prendre systématiquement les mêmes décisions.

Un tel algorithme est naturellement anytime : une séquence d'actions (et donc une

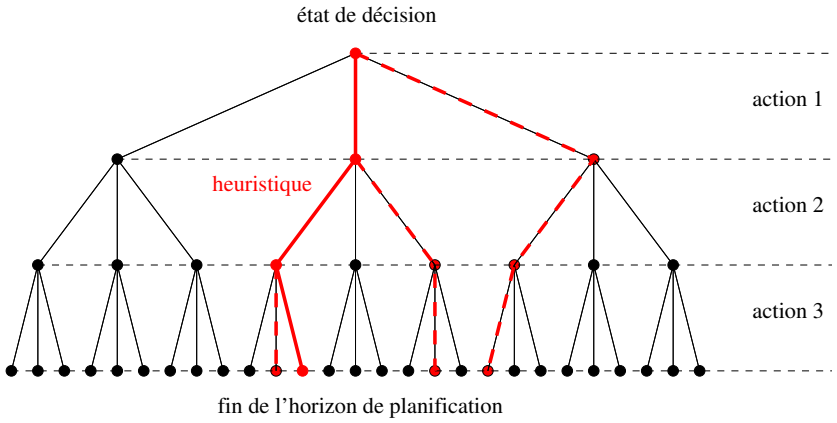


FIG. 5.4: Construction d'un plan par l'algorithme glouton stochastique itéré

première solution) est disponible dès la première recherche gloutonne, ce qui est rapide. La solution est améliorée à chaque fois qu'une recherche gloutonne construit un plan dont le gain associé est meilleur que le meilleur gain produit jusqu'alors. Cependant un tel algorithme n'offre aucune garantie en terme d'optimalité.

3.1.2 Heuristique et aspect stochastique

L'heuristique qui guide l'algorithme de planification implémenté dans le logiciel de vol est basée sur la règle de décision 1 (cf section 2.2.2). On introduit alors une règle de décision 1 bruitée (SDR1 pour Stochastic Decision Rule 1) inspirée de DR1 et définie par trois paramètres p, q et r. Comme le décrit la figure 5.5, le rôle de ces trois paramètres est d'introduire un caractère aléatoire dans le résultat rendu par la règle SDR1.

Le paramètre p représente la probabilité de choisir une action de même niveau de priorité que l'action qui serait choisie par la règle de décision DR1[2].
Le paramètre q représente, dans le cas où le type d'action "observation" a été choisi et que plusieurs zones à observer sont possibles, la probabilité de choisir la même zone à observer que celle qui serait choisie par la règle de décision DR1.
Le paramètre r représente, dans le cas où le type d'action "observation" a été choisi, que plusieurs zones à observer sont possibles, et que la zone choisie n'est pas celle qui serait choisie par la règle de décision DR1, la probabilité de choisir la zone la moins couverte par les nuages.

Chaque recherche gloutonne construit alors un plan d'actions à partir de l'état de décision, en prenant des décisions successives correspondant aux décisions rendues par la

[2]Dans le cas de la règle de décision 1, les différents types d'actions sont classés par ordre de priorités décroissantes : manœuvre orbitale, télédéchargement, détection de nuages, observation, rechargement des batteries, puis pointage géocentrique.

règle de décision 1 bruitée (cf figure 5.5). Le réglage des paramètres p, q et r sera étudié au chapitre 7.

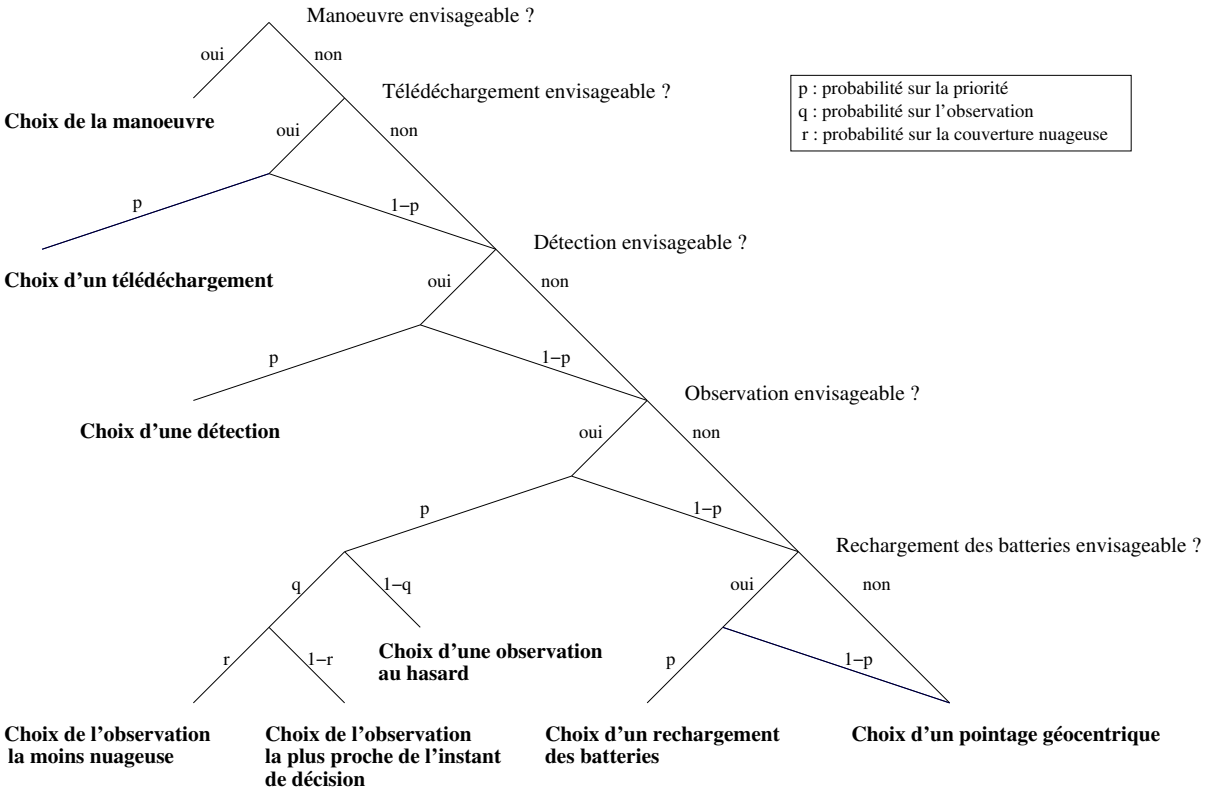


FIG. 5.5: Comportement de la règle de décision 1 bruitée, SDR1

Le code source partiel de l'algorithme glouton stochastique itéré qui a été développé est disponible à l'annexe D.

3.1.3 Cas particulier des actions de détections

À partir du moment où les ressources du satellite sont suffisantes, une action de détection de la couverture nuageuse est envisageable à tout instant. Afin de simplifier le problème de planification, nous avons développé, dans un premier temps, une méthode de détermination a priori de l'ensemble des actions de détection nécessaires sur un certain horizon de planification. On se reportera à l'annexe C pour en connaître les détails.

Dans un second temps, les actions de détection de la couverture nuageuse ont été traitées comme les autres actions (avec des préconditions et des effets) et la possibilité d'insérer à tout moment une telle action dans le plan d'activités du satellite a été laissée à la convenance du planificateur.

3.2 Initialisation d'une délibération

La tâche délibérative est réinitialisée et relancée par la tâche réactive, à chaque prise de décision et à chaque changement d'état pertinent du système. La part d'aléatoire de l'algorithme d'optimisation ne lui interdit pas de rendre des délibérations (au moins les premières) de moins bonne qualité que la décision qui aurait été fournie par une simple règle de décision. Afin de palier à ce problème, nous forçons la tâche délibérative à rendre comme première délibération, la première action du plan construit uniquement à partir de la règle de décision implémentée dans la tâche réactive. On assure ainsi que les délibérations rendues par la tâche délibérative seront toutes au moins d'aussi bonne qualité qu'une simple décision réactive.

Dans le cas où la tâche délibérative est relancée suite à une prise de décision, il serait dommage de redémarrer l'optimisation de la prochaine décision, sans tenir compte du raisonnement réalisé par la tâche délibérative précédente. Ainsi nous forçons la tâche délibérative à rendre comme première délibération, non plus la première action du plan construit uniquement à partir de la règle de décision, mais la première action du plan de meilleure qualité parmi le plan construit uniquement à partir de la règle de décision, et le meilleur plan précédent tronqué de l'action qui vient d'être engagée, et éventuellement complété jusqu'à la fin du nouvel horizon de planification, en appliquant la règle de décision.

4 Comportement temporel

Le comportement temporel de l'architecture réactive - délibérative est le suivant :
La tâche réactive :

- reçoit les stimuli de l'environnement correspondant aux changements d'état du système, et filtre les changements pertinents (ceux qui nécessitent un abandon et une relance de la tâche délibérative) ;
- à chaque changement d'état pertinent :
  - calcule la prochaine échéance ;
  - abandonne et relance la tâche délibérative.
- à chaque échéance, décide de l'action à engager, en prenant en compte l'état courant du système et soit (1) les délibérations fournies par la tâche délibérative si elles existent, soit (2) une décision par défaut calculée par la tâche réactive elle-même.

La tâche délibérative :

- exécute une tâche de raisonnement parallèlement à la tâche réactive qui la contrôle ;
- est abandonnée et relancée par la tâche réactive, à chaque changement d'état pertinent ;
- fournit aussi vite que possible, des délibérations à la tâche réactive.

La figure 5.6 représente un exemple d'exécution typique. Le temps évolue de haut en bas. Un évènement *changement* déclenche le calcul d'une échéance émise par l'in-

termédiaire de l'évènement *prochÉchéance*, et l'exécution d'une nouvelle tâche délibérative informée de l'état courant, de l'action courante et de la prochaine échéance. Avant la fin de l'échéance, la tâche délibérative émet deux délibérations successives. Puis l'évènement *échéance* déclenche la prise d'une décision, l'*engagement* de cette décision, le calcul d'une nouvelle échéance, l'émission de l'évènement *prochÉchéance* et la relance de la tâche délibérative.

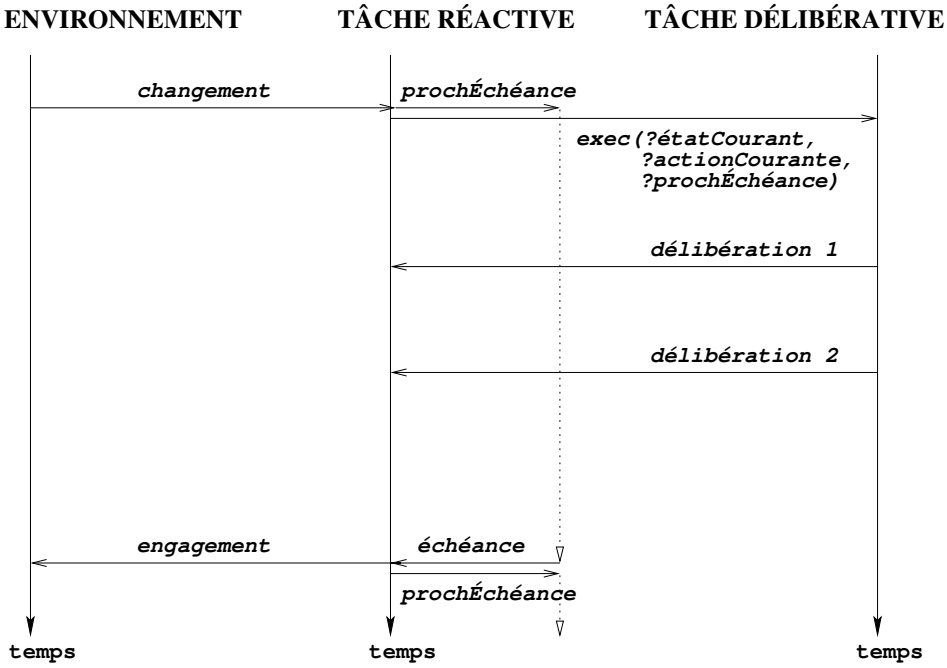


FIG. 5.6: Comportement temporel

Ces travaux ont donné lieu à une communication internationale [Beaumet *et al.*, 2008a] à la conférence iSAIRAS'08 (international Symposium on Artificial Intelligence, Robotics and Automation in Space).

Chapitre 6

Simulateur de mission - Environnement expérimental

1 Motivations

Une fois les mécanismes de raisonnement définis et implémentés au sein de l'architecture réactive - délibérative, nous avons eu besoin dans un premier temps de régler, puis dans un second temps de tester le comportement de cette architecture et des algorithmes d'optimisation. Nous avons donc développé à cet effet un environnement de simulation de mission satellitaire d'observation de la Terre.

2 Conception

2.1 Les grandes lignes

L'outil développé devait nous permettre d'une part de simuler l'évolution d'une mission satellitaire d'observation de la Terre, et d'autre part de tester au cours de simulations hors ligne et en ligne, le comportement du logiciel de vol mis en place.

2.2 Couplage entre le simulateur et le logiciel de vol

L'application se présente donc en deux parties distinctes dont les interactions sont représentées sur la figure 6.1. On reconnaît sur la partie droite de la figure, le logiciel de vol décrit au chapitre 5 et composé d'une partie réactive et d'une partie délibérative. La partie réactive informe la tâche délibérative des différents *changements d'état* et de la *prochaine échéance*, et reçoit en contrepartie des *délibérations*. Ces deux tâches basent leurs raisonnements (décisions réactives, optimisation, calcul d'échéance, trajectoires en

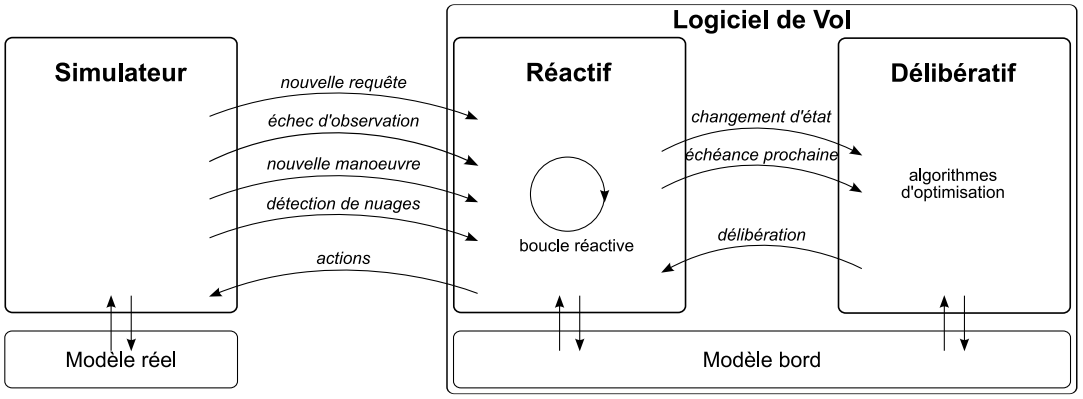


FIG. 6.1: Couplage entre le simulateur et le logiciel de vol

attitude, etc.) sur un modèle bord du satellite, de la mission et de ses objectifs, embarqué à bord de l'engin.

Ce logiciel de vol viendra "se brancher" sur le simulateur de la mission, décrit sur la partie gauche de la figure et qui lui, se base sur un second modèle de la réalité qui peut être différent du modèle utilisé à bord du satellite : modélisation différente, éventuellement plus détaillée, du satellite, de la mission et de ses objectifs.

Les messages échangés entre le logiciel de vol et le simulateur de la réalité sont de cinq types :

- le simulateur simule l'envoi de *nouvelles requêtes* d'observation, des stations sol vers le satellite lors des fenêtres de visibilité bord - sol ;
- il simule l'*échec* de la réalisation de certaines observations du à une défaillance de l'instrument d'observation ;
- il simule la génération, par le Contrôle d'Orbite Autonome du satellite, de *nouvelles manœuvres* orbitales à réaliser pour replacer le satellite sur son orbite de référence ;
- il simule l'envoi au satellite d'informations sur l'état de la couverture nuageuse courante, suite à une action de *détection* ;
- enfin, le simulateur reçoit de la part du logiciel de vol du satellite des évènements d'engagement d'*actions* et simule leurs effets sur le monde réel.

2.3 Diagramme de classes simplifié

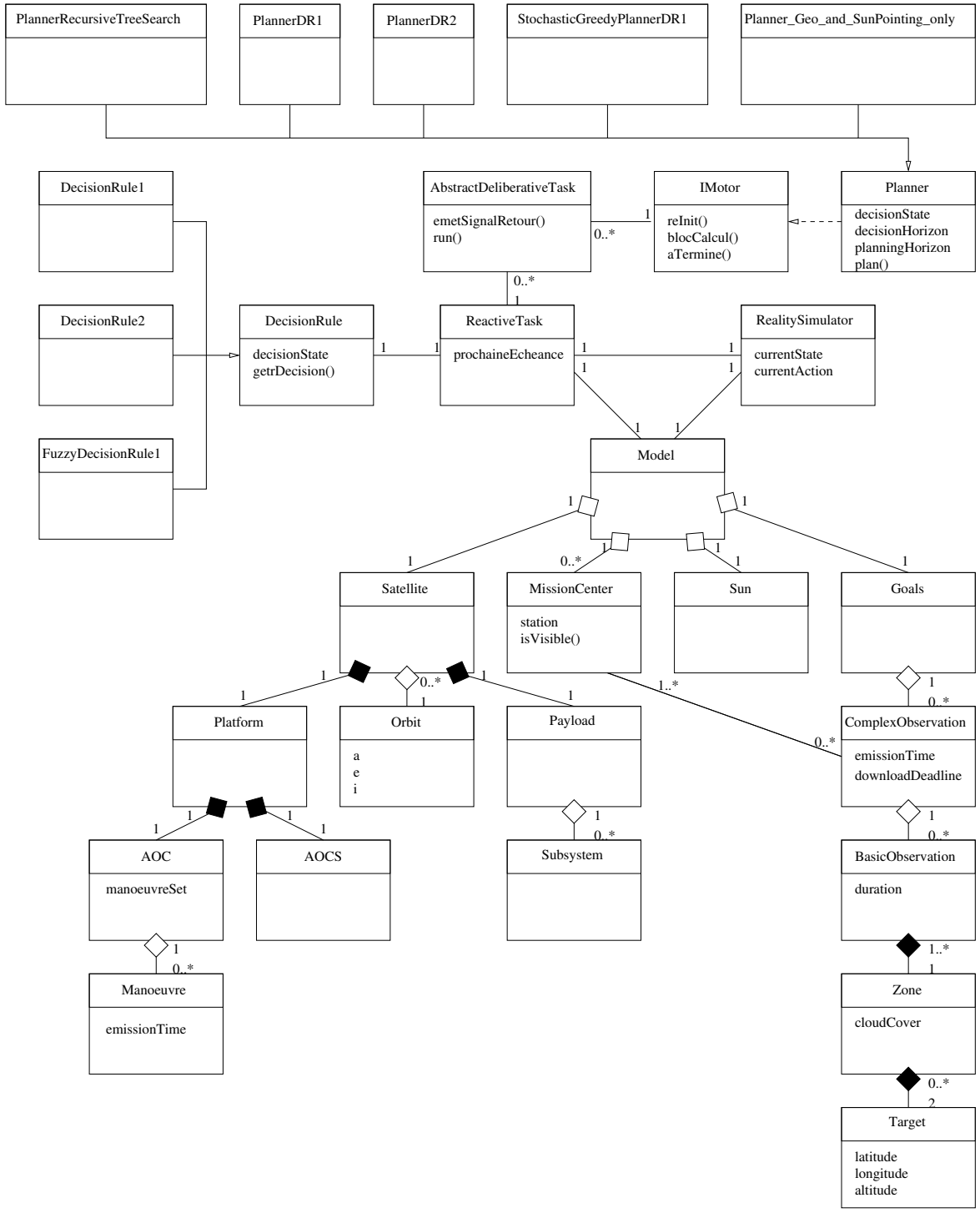


FIG. 6.2: Diagramme de classes simplifié

3 Modélisation

3.1 Modélisation cinématique

3.1.1 Définitions

L'unité astronomique, UA, correspond à la distance moyenne entre la Terre et le Soleil ; soit approximativement la longueur du demi-grand axe de l'orbite terrestre.

$$1 \text{ UA} = 149597870,691 \text{ km} \pm 30 \text{ m}$$

e_0 et ϵ_0 définissent respectivement l'excentricité et l'obliquité de l'orbite terrestre à la date $J2000$ (1^{er} janvier 2000 à 12 heures).

$a = 1.0000002$ UA définit le demi-grand axe de l'ellipse décrite par le mouvement de la Terre autour du Soleil.

La période de temps qui sépare deux passages consécutifs au périhélie s'appelle l'*année anomalistique*, notée τ_{ano}. Elle est estimée actuellement à :

$$\tau_{ano} = 365.2596445 \text{ jours}$$

Le mouvement moyen n_{ano} de la Terre autour du Soleil est alors défini par :

$$n_{ano} = \frac{360}{\tau_{ano}} = 0.985600258\,^{\circ}/\text{jour}$$

La période de temps qui sépare deux passages consécutifs de la Terre au point vernal, γ (qui précessionne par rapport aux étoiles), s'appelle l'*année tropique*. Elle est estimée actuellement à :

$$\tau_{tro} = 365.24218967 \text{ jours solaires}$$

Le mouvement moyen n_{tro} relatif au passage au point vernal est alors défini par :

$$n_{tro} = \frac{360}{\tau_{tro}} = 0.98564735\,^{\circ}/\text{jour}$$

3.1.2 Cinématique Terre - Soleil

Distance Terre - Soleil Afin de simplifier la modélisation, nous avons fait quelques approximations raisonnables :

1. La lumière reçue au voisinage de la Terre à un instant t provient d'une position occupée par le Soleil $n = \frac{\texttt{distance Terre - Soleil}}{\texttt{vitesse de la lumière}} \simeq 500$ secondes plus tôt. La distance Terre - Soleil étant tellement grande, nous avons choisi de négliger ce retard du à la vitesse de propagation de la lumière.
2. Lorsque l'on s'intéresse au mouvement de la Terre autour du Soleil, on s'intéresse plus exactement au mouvement du barycentre du système Terre - Lune. L'attraction de la Lune entraîne en effet une perturbation du mouvement de la Terre sur son orbite. Cependant cette perturbation est très faible (écart de l'ordre de $\frac{\texttt{distance centre Terre - barycentre}}{\texttt{demi-grand axe orbite terrestre}} \simeq 0.003\%$ par rapport à l'orbite de référence), et sera négligée par la suite.
3. L'excentricité e de l'orbite terrestre sera supposée constante :

$$e = e_0 = 0.016709114$$

Sous ces hypothèses, nous modélisons le mouvement de la Terre autour du Soleil de la même manière que dans le chapitre 4. L'anomalie moyenne de la Terre à un instant t, $M(t)$, est donnée par :

$$M(t) = n_{ano}(t - t_{po}) = M_0 + n_{ano} \cdot t \text{ avec } M_0 = -n_{ano} \cdot t_{po}$$

où t_{po} est la date de passage au périhélie de l'orbite. Si t_{po} correspond à la date du 4 janvier 2000 à 0h 15mn et 24 s, on a :

$$M_0 \simeq -\frac{360}{\tau_{ano}} \times 2.5107 = -2.47456\,^\circ$$

L'anomalie vraie de la Terre à un instant t, $\nu(t)$, est donnée par :

$$\nu(t) = M(t) + C$$

où la différence C entre les anomalies vraie et moyenne vaut :

$$C = (2e - \frac{1}{4}e^3)\sin(M) + (\frac{5}{4}e^2 - \frac{11}{24}e^4)\sin(2M) + \frac{13}{12}e^3 \cdot \sin(3M) + \frac{103}{96}e^4 \cdot \sin(4M)$$

L'équation de l'ellipse permet alors d'en déduire la distance Terre - Soleil à tout instant t, $\rho(t)$:

$$\rho(t) = \frac{a(1 - e^2)}{1 + e \cdot \cos(\nu(t))}$$

Position du Soleil Comme dans le paragraphe précédent, il est possible de faire quelques approximations :

1. L'obliquité ϵ de l'orbite terrestre sera supposée constante :

$$\epsilon = \epsilon_0 = 23.43928\,^\circ$$

2. Au regard de sa période (un peu plus de 18 ans) par rapport aux durées de simulation que nous utilisons (quelques jours au maximum) et de son amplitude (17,2 secondes d'arc au maximum), le mouvement de nutation de l'axe de rotation de la Terre autour de sa position moyenne peut être négligé.

La longitude écliptique vraie du Soleil à un instant t est donnée par :

$$\Lambda_\odot(t) = \Lambda_{\bar{\odot}_{d_0}} + n_{tro} \cdot t + C$$

où $\Lambda_{\bar{\odot}_{d_0}}$ représente la longitude écliptique du Soleil dynamique à la date $t = 0$.

La position du Soleil à tout instant t, dans le repère céleste $\mathcal{R}_C$ est alors donnée par :

$$\overrightarrow{T\odot}(t) = \begin{bmatrix} x_\odot(t) \\ y_\odot(t) \\ z_\odot(t) \end{bmatrix}^{R_C} = \rho(t) \begin{bmatrix} \cos(\Lambda_\odot(t)) \\ \sin(\Lambda_\odot(t)) \cdot \cos(\epsilon) \\ \sin(\Lambda_\odot(t)) \cdot \sin(\epsilon) \end{bmatrix}^{R_C}$$

3.1.3 Satellite

L'orbite d'un satellite est définie par l'ensemble de ses paramètres orbitaux : a, e, i, ω, Ω, n et t_P décrits à la section 1.2 du chapitre 4. L'état cinématique d'un satellite à un instant t est alors défini par sa position dans l'espace par rapport au centre de la Terre, sa vitesse de translation sur son orbite et son attitude.

3.2 Modélisation du satellite

3.2.1 Plate-forme

La plate-forme du satellite est principalement composée :

– d'un système de Contrôle d'Orbite Autonome (COA) : son rôle est de garder le satellite sur son orbite de référence. La modélisation de cet élément est donc réduite à un ensemble de manœuvres orbitales qui seront régulièrement envoyées du COA au logiciel de vol du satellite, au cours de la mission ;

– d'un Système de Contrôle d'Attitude et d'Orbite (SCAO) : une des caractéristiques cinématiques importantes du satellite considéré est que ses mouvements en attitude selon ses trois axes de rotation sont indépendants. Les actionneurs utilisés sont donc soit trois roues à réaction, soit trois paires d'actionneurs gyroscopiques. Les moments cinétiques selon les axes de roulis, tangage et lacet sont respectivement de 45, 45 et

20 N.m.s, et le satellite développe respectivement des couples de 7, 7 et 6 N.m selon ces trois axes. Enfin, le module maximum de la vitesse de rotation instantanée de l'engin est limitée à 10 rd.s^{-1} ;
- de batteries d'une capacité maximale de 5800 Wh ;
- d'une mémoire de masse de 600 Gbits ;
- d'un réservoir d'ergol d'une contenance de 80 litres ;
- et de générateurs solaires dont le taux de consommation d'énergie vaut 1 Wh.s^{-1} et dont le taux de production maximum (lorsque les panneaux sont perpendiculaires aux rayons du Soleil) d'énergie est $\tau_{max} = 2$ Wh.s^{-1}. Si $\vec{u}_\odot$ et $\vec{z}_S$ sont respectivement le vecteur unitaire dans la direction Terre - Soleil et le vecteur unitaire désignant l'axe de lacet du satellite (orthogonal au plan des panneaux solaires), le taux τ de production d'énergie en fonction de l'orientation des panneaux solaires est donné par (voir figure 6.3) :

$$\begin{aligned} \tau &= \begin{cases} \tau_{max} \cdot \langle \vec{u}_\odot \cdot \vec{z}_S \rangle \text{ si } \langle \vec{u}_\odot \cdot \vec{z}_S \rangle \geq 0 \\ 0 \text{ sinon} \end{cases} \\ &= \begin{cases} \tau_{max} \cdot \cos(\alpha) \text{ si } \cos(\alpha) \geq 0 \\ 0 \text{ sinon} \end{cases} \end{aligned}$$

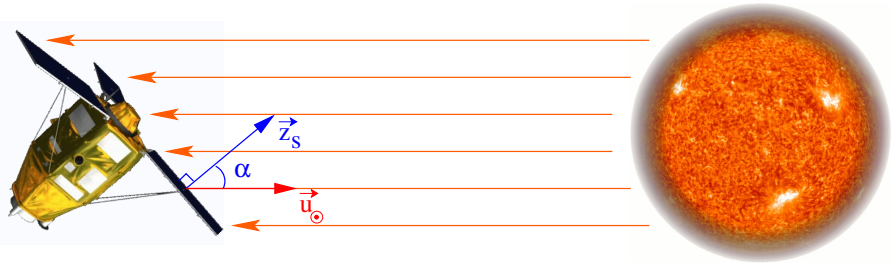


FIG. 6.3: Production d'énergie par les générateurs solaires

Enfin, l'inertie du satellite selon ses axes de roulis et tangage est de 850 m^2.kg, tandis que son inertie en lacet est de 750 m^2.kg.

3.2.2 Charge utile

De son côté, la charge utile du satellite se compose :
- d'un instrument d'observation dont le taux de consommation d'énergie vaut 1 Wh.s^{-1} ;
- d'un instrument de détection de la couverture nuageuse dont le taux de consommation d'énergie vaut 1 Wh.s^{-1}. Cet instrument nécessite une rotation du satellite de 30 $^\circ$ en tangage, pour détecter la couverture nuageuse en avant de l'engin ;
- et d'une antenne de télédéchargement des données dont le taux de consommation d'énergie vaut 1 Wh.s^{-1}, et le débit maximum est de 450 Mbits par secondes.

3.2.3 Actions du satellite

Production d'énergie au cours d'une action La production d'énergie p_{en} au cours d'une action a est fonction de la trajectoire en attitude suivie par le satellite au cours de l'action a. Cette production d'énergie est estimée en multipliant le taux de production d'énergie moyen (moyenne entre les taux de production dans les attitudes de début et de fin de l'action a) des panneaux solaires au cours de l'action a par la durée de a.

$$p_{en}(a) = \frac{\tau(\texttt{attitude\_début}(a)) + \tau(\texttt{attitude\_fin}(a))}{2} \times \texttt{durée}(a)$$

Manœuvres orbitales Les actions de manœuvre orbitale sont déterminées par le Contrôle d'Orbite Autonome du satellite. Elles sont définies par une attitude de départ et une durée de poussée précises. Elles ont donc ainsi des consommations en énergie et en ergol bien déterminées.

3.3 Modèle météorologique

On rappelle que le satellite d'observation considéré est équipé d'un instrument de détection de la couverture nuageuse afin de diminuer le nombre d'échecs lors de l'observation de zones couvertes en nuages (aujourd'hui près de 80% des observations d'un satellite Spot sont inexploitables à cause des nuages).

Afin de nous permettre d'évaluer l'intérêt d'utiliser un tel instrument à bord du satellite, le simulateur de mission que nous avons développé devait être capable de traiter l'évolution de la couverture nuageuse terrestre. Trois types de modèles météorologiques ont été nécessaires :

- un modèle d'évolution de la couverture nuageuse réelle à la surface de la Terre ;
- un modèle des prévisions d'évolution de la couverture nuageuse, utilisé dans le centre de contrôle qui transmet au satellite les prévisions de couverture nuageuse à chacun de ses passages en visibilité ;
- et un modèle de la couverture nuageuse connue à bord du satellite ; modèle sur lequel se base le satellite pour raisonner.

3.3.1 Couverture nuageuse réelle

Un quadrillage du planisphère, de 144 par 72 unités, représenté sur la figure 6.4, permet de représenter l'état de la couverture nuageuse à la surface de la Terre.

À chaque case du quadrillage est associée une valeur réelle comprise entre 0 et 1 représentant le pourcentage de couverture nuageuse au dessus de la zone (0 : aucun nuage au dessus de la zone ; 1 : les nuages recouvrent la totalité de la zone). On fait donc l'hypothèse que la couverture nuageuse est homogène sur une même case. Cette hypothèse

est réaliste car la surface terrestre maximale d'une case ne dépasse pas[1] :

$$(\frac{2\pi}{144}R_T) \cdot (\frac{\pi}{72}R_T) \simeq 77000 \text{ km}^2$$

où R_T désigne le rayon terrestre moyen.

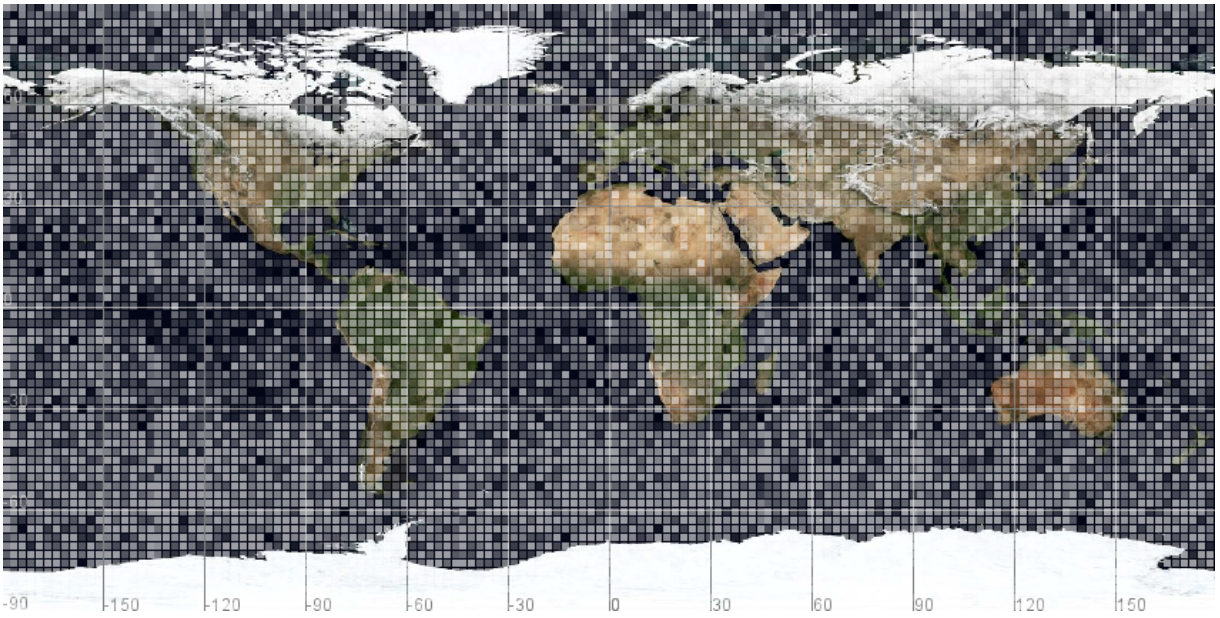


FIG. 6.4: Quadrillage de la couverture nuageuse à la surface de la Terre

Un état de la couverture nuageuse mondiale réelle est donc représenté par une grille de $144 \times 72 = 10368$ cases. Pour modéliser l'évolution de la couverture nuageuse mondiale, nous sommes partis de statistiques de couvertures nuageuses mensuelles moyennes à la surface du globe établies entre les années 1981 et 1991[2].

Pour chaque case de la grille, nous avons extrapolé la valeur de sa couverture nuageuse réelle, en suivant une loi gaussienne d'espérance la moyenne des valeurs mensuelles moyennes de couverture nuageuse de la case en question et de ses cases adjacentes (pour simuler une certaine continuité de la couverture nuageuse) et d'écart type 30%, et cela pour chaque tranche de trois heures du mois en cours. Nous avons ainsi obtenu une succession de grilles représentant l'évolution de la couverture nuageuse réelle à la surface de la Terre pendant une année, avec un pas de trois heures[3]. La figure 6.5 résume la construction de ce modèle d'évolution de la couverture nuageuse réelle.

[1]Ce calcul représente la surface d'une case à l'équateur, là où la surface est maximale. À titre de comparaison, la superficie de la France métropolitaine avoisine les 550000 km^2.

[2]Ces données ont été récupérées sur le site internet de l'International Satellite Cloud Climatology

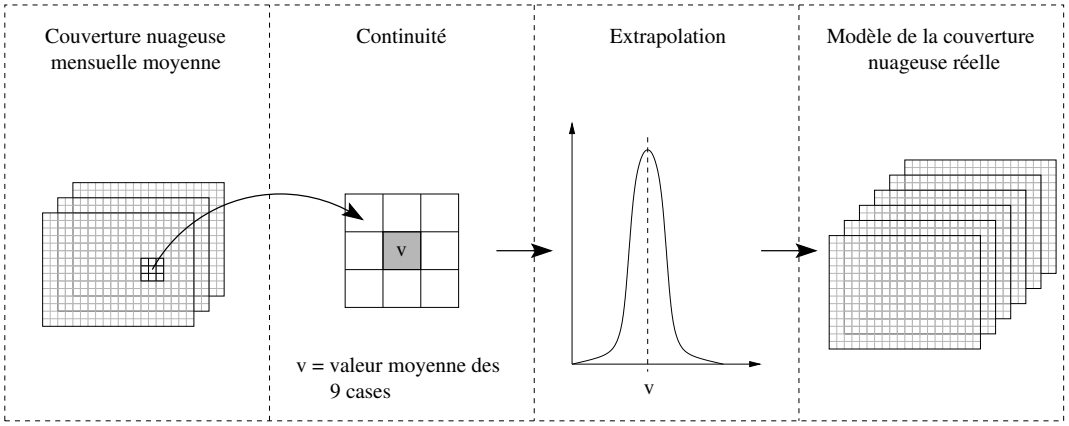


FIG. 6.5: Construction du modèle d'évolution de la couverture nuageuse réelle

3.3.2 Prévisions de couverture nuageuse

Actuellement, les centres de contrôle utilisent des prévisions météorologiques pour construire, au sol, les plans d'activités des satellites d'observation. Dans le contexte de la mission d'observation par un satellite autonome, la couverture nuageuse en avant de l'engin, peut être détectée par le satellite pour orienter ses décisions à court terme, mais des prévisions météorologiques fournies par le sol restent très utiles pour des raisonnements à plus long terme.

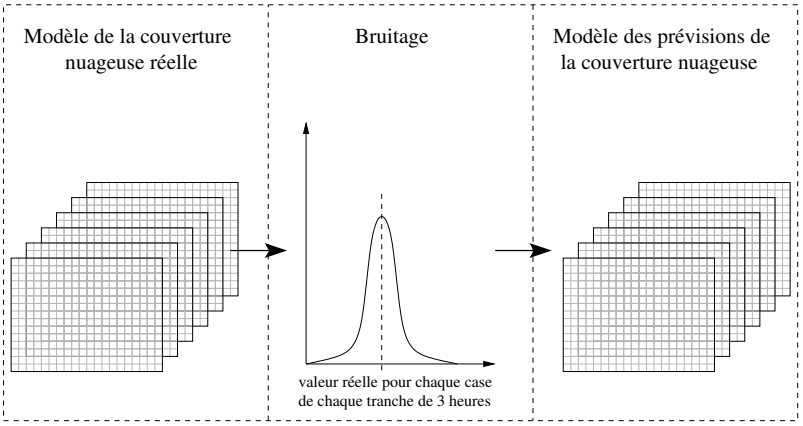


FIG. 6.6: Construction du modèle des prévisions d'évolution de la couverture nuageuse

Project (ISCCP)(http ://isccp.giss.nasa.gov/index.html).

[3]Comme nous ne disposons pas de modèle d'évolution plus précis de la couverture nuageuse mondiale, on fait ici l'hypothèse que la couverture nuageuse évolue instantanément toutes les trois heures et reste constante entre deux évolutions.

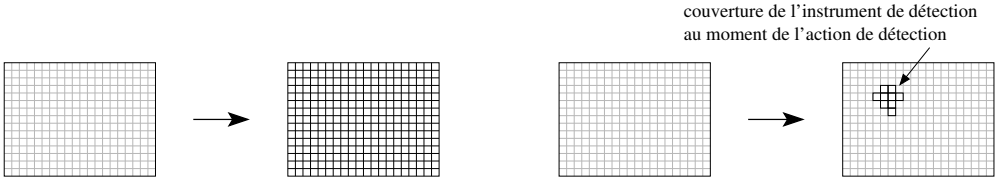
C'est pourquoi nous avons développé un modèle des prévisions d'évolution de la couverture nuageuse qui permet au centre de contrôle d'informer le satellite sur l'état futur de la couverture nuageuse, à chaque fois que celui-ci passe en visibilité du centre.

Ce modèle des prévisions d'évolution de la couverture nuageuse a été obtenu en partant du modèle d'évolution de la couverture nuageuse réelle, et en choisissant pour chaque case de chaque grille correspondant à chaque tranche horaire de trois heures, une valeur respectant une loi gaussienne d'espérance la valeur de la couverture réelle et d'écart type faible (10%). La construction de se modèle est résumée sur la figure 6.6.

3.3.3 Couverture nuageuse bord

Le satellite ne dispose pas, à bord, d'un modèle d'évolution de la couverture nuageuse. Il se contente de maintenir à jour une croyance sur la couverture nuageuse mondiale courante. Celle-ci se représente sous la forme d'une seule grille de $144 \times 72 = 10368$ cases. Initialisée avec les valeurs mensuelles moyennes de la couverture nuageuse, cette grille est mise à jour :

- entièrement lors de la réception d'une information de prévision de la couverture nuageuse, en provenance d'un centre de contrôle : toutes les cases de la grille sont mises à jour (voir figure 6.7(a)) ;
- partiellement suite à une action de détection de la couverture nuageuse en avant du satellite : seules les cases appartenant à la zone couverte par l'instrument de détection sont mises à jour (voir figure 6.7(b)).



(a) Prévisions météorologiques (b) Détection

FIG. 6.7: Mise à jour de la couverture nuageuse bord

3.4 Modélisation des objectifs

Les objectifs de la mission, c'est-à-dire les observations à réaliser, sont modélisés par un ensemble d'observations complexes. Chaque observation complexe est définie par une ou plusieurs observations élémentaires, un statut de réalisation ("nonCommencee", "enCours", "realisee" ou "teledechargee"), une date d'émission par le centre de contrôle et une date limite de télédéchargement au delà de laquelle l'observation complexe devient obsolète. Une observation complexe permet de regrouper un ensemble d'observations

élémentaires couvrant une grande surface au sol (voir figure 6.8), un couple (respectivement un triplet) d'observations stéréoscopiques (respectivement tri-stéréoscopiques) de la même zone au sol, ou bien un ensemble d'observations élémentaires de la même zone au sol à des dates différentes (pour le suivi de l'évolution de la zone).

Une observation élémentaire représente l'observation d'une bande au sol de largeur inférieure ou égale à la fauchée de l'instrument d'observation du satellite. Chaque observation élémentaire est modélisée par des fenêtres de visibilité par les instruments d'observation et de détection du satellite, par un mode d'observation (normal ou stéréoscopique), des contraintes de qualité, une vitesse de balayage au sol, des consommations de ressources (énergie et mémoire) et la zone géographique à observer.

Enfin, une zone géographique est représentée par une bande rectangulaire définie par deux cibles au sol, et de largeur égale à la largeur de la fauchée de l'instrument d'observation du satellite.

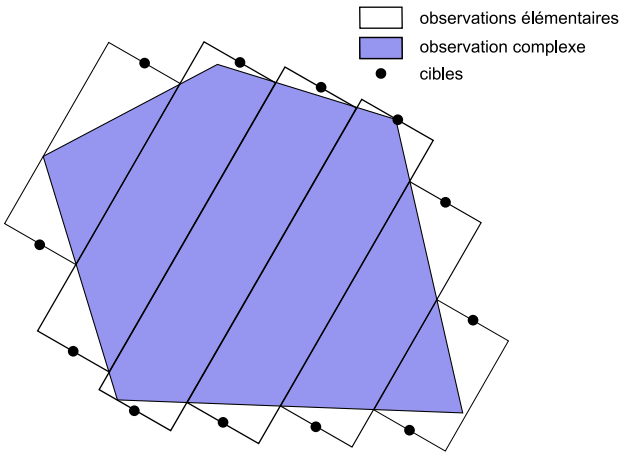


FIG. 6.8: Découpage d'une zone au sol en six bandes élémentaires

4 Interface graphique

4.1 Fenêtre principale

L'interface graphique de l'application développée se compose d'une fenêtre principale et de plusieurs fenêtres de dialogue avec l'utilisateur. La figure 6.9 représente la fenêtre principale de l'application ainsi que la vue en trois dimensions du satellite. Elle se décompose en deux parties principales :

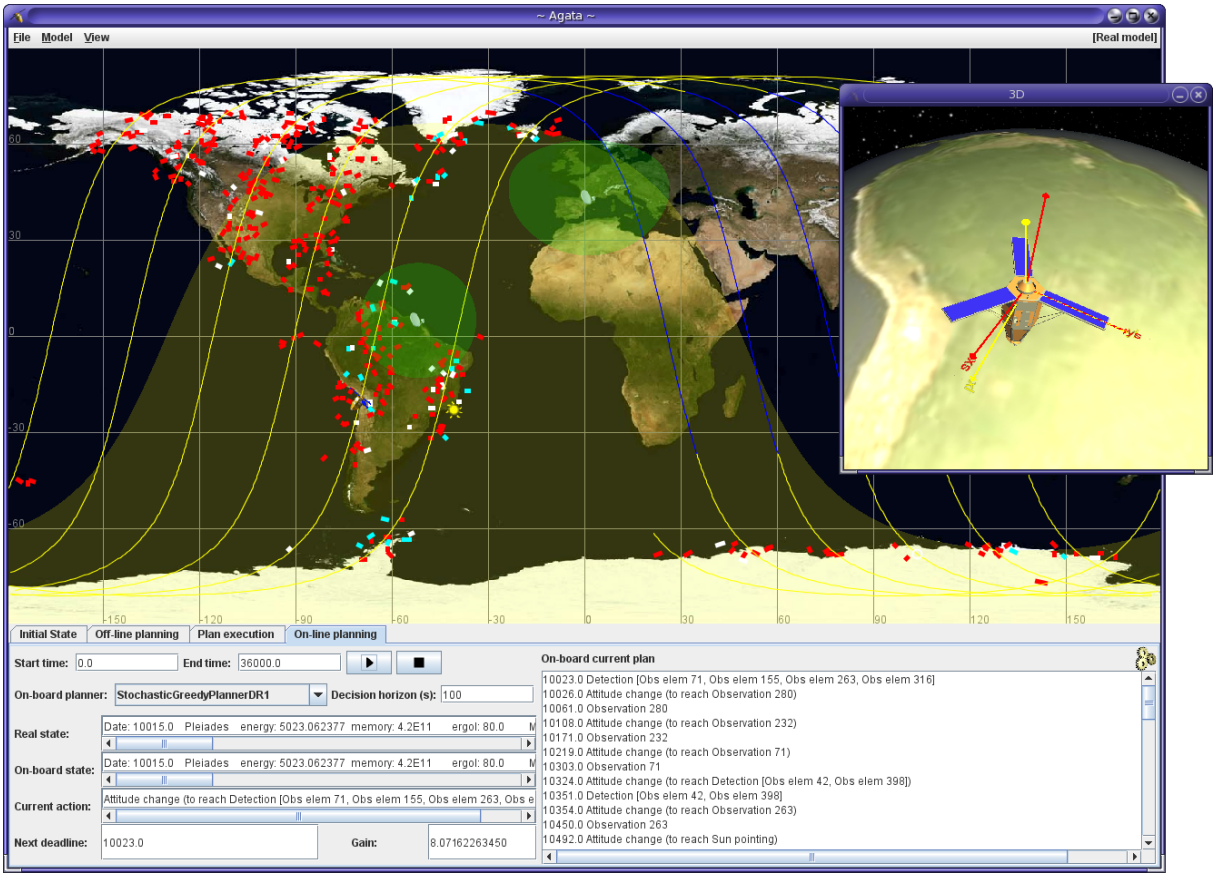


FIG. 6.9: Interface graphique

- dans la partie supérieure, un planisphère montre une représentation de l'état courant du monde réel, ou bien une représentation de l'état courant du système tel qu'il est perçu par le satellite ;
- et dans la partie inférieure, une série de quatre onglets permettent à l'utilisateur de modifier l'état courant du satellite, lancer des planifications hors ligne, simuler l'exécution des plans construits, ou bien lancer une exécution temps-réel d'un scénario d'observation de la Terre en utilisant des mécanismes de décision en ligne à bord du satellite.

Une barre de menu permet de gérer les différents modèles utilisés pour la simulation ainsi que l'affichage des différents composants.

4.2 Affichage

4.2.1 Éléments graphiques

Les différents éléments du modèle représentés sur le planisphère sont les suivants :
- le satellite d'observation et son orbite (de couleur jaune lorsque le satellite est éclairé, bleue lorsqu'il est en éclipse) ;
- la position du Soleil ainsi que la surface éclairée de la Terre (en jaune) ;
- les centres de mission (représentés par des paraboles) et leurs cônes de visibilité associés (en vert) ;
- les observations élémentaires à réaliser, représentées par des bandes de couleur
  - blanche lorsque leur requête associée n'a pas encore été émise par le centre de contrôle ;
  - rouge vif lorsque leur requête associée a été envoyée au satellite ;
  - rouge sombre lorsque leur couverture nuageuse a récemment été détectée par le satellite ;
  - cyan lorsqu'elles sont réalisées ;
  - bleu lorsqu'elles sont télédéchargées.
- les manœuvres orbitales représentées par des cercles de couleur
  - blanche lorsqu'elles n'ont pas encore été générées par le Contrôle d'Orbite Autonome du satellite ;
  - rouge lorsqu'elles sont connues à bord du satellite ;
  - cyan lorsqu'elles sont réalisées.
- la couverture nuageuse, représentée par un quadrillage de couleur blanche, plus ou moins transparent, du planisphère.

Une vue en trois dimensions permet de visualiser les mouvements du satellite sur son orbite et en attitude. Les repères satellite (en rouge) et orbital local (en jaune) sont également représentés.

4.2.2 Gestion de l'affichage

Le menu *View* de l'application permet à l'utilisateur de gérer l'affichage, en l'activant ou le désactivant. Une désactivation de l'affichage permet de diminuer les temps de calculs de l'application. Il lui permet également de choisir les différents éléments à afficher parmi les éléments graphiques listés à la section 4.2.1, ainsi que de basculer de l'affichage du modèle réel (*[Real model]*) à celui du modèle bord (*[On-board model]*) représentant l'état du système vu par le satellite.

5 Fonctionnalités

Dans cette section nous décrivons les différentes fonctionnalités offertes par l'outil développé.

5.1 Gestion des scénarios

L'interface graphique de l'outil permet à l'utilisateur de définir facilement des scénarios de simulation. Le menu *model* permet de créer, éditer, ou supprimer des observations complexes, des observations élémentaires, des centres de mission et des manœuvres orbitales. Il permet également de définir les dates de début et de fin du scénario, ainsi que les paramètres orbitaux du satellite. Un modèle réel ou bord peut aussi être sauvegardé ou chargé directement à partir d'un fichier d'entrée dont le format est décrit à la section 6. Un dernier sous-menu permet à l'utilisateur de générer automatiquement et aléatoirement des scénarios de simulation en ne définissant que la durée du scénario, les paramètres orbitaux du satellite, le nombre de manœuvres orbitales et le nombre d'observations à réaliser. Ce menu permet de créer facilement des scénarios réalistes que l'utilisateur pourra jouer à sa guise.

5.2 Définition et contrôle de l'état initial

La partie inférieure de la fenêtre principale de l'interface graphique de l'environnement de simulation, est composée de quatre onglets offrant chacun une fonctionnalité. Le premier onglet, intitulé *Initial State* et représenté sur la figure 6.10, s'intéresse à la définition de l'état du satellite à la date de début d'une planification ou d'une simulation d'exécution.

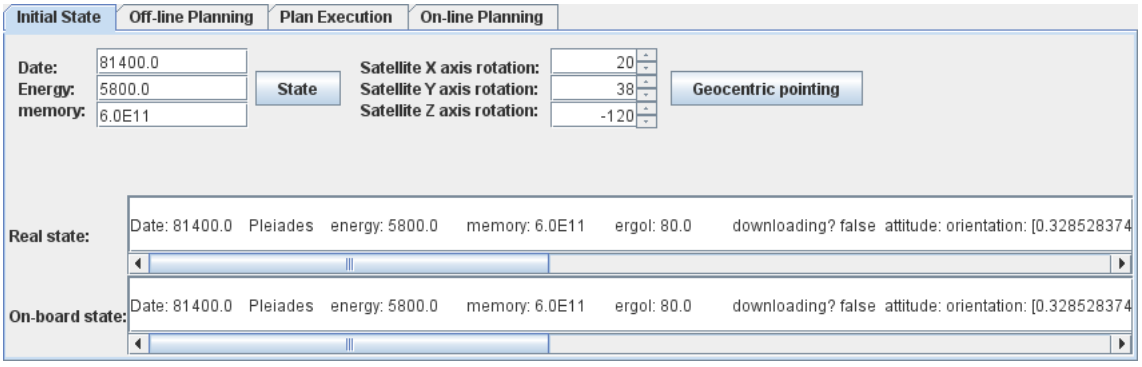


FIG. 6.10: Onglet *Initial State* de l'environnement de simulation

L'utilisateur peut y définir la date de début de la simulation (ou planification), les niveaux d'énergie et de mémoire libre initiaux du satellite, ainsi que son attitude initiale (angles de roulis, tangage et lacet). L'état réel de l'environnement et l'état estimé à bord du satellite sont représentés dans la partie inférieure de l'onglet.

5.3 Planification hors ligne

Un deuxième onglet, intitulé *Off-line Planning* et représenté sur la figure 6.11, permet de tester différents planificateurs sur le scénario courant. Il suffit à l'utilisateur de définir

les horizons de planification et de décision, et le planificateur qu'il souhaite tester. La planification s'effectue alors hors ligne, comme si elle était réalisée au sol, dans un centre de contrôle, et une zone de texte permet à l'utilisateur de suivre en temps-réel, la construction du plan. En fonction du planificateur testé, certains paramètres peuvent alors s'afficher en cours de planification, comme le nombre d'états parcourus par une recherche arborescente ou le nombre de descentes effectuées par une recherche gloutonne stochastique itérée.

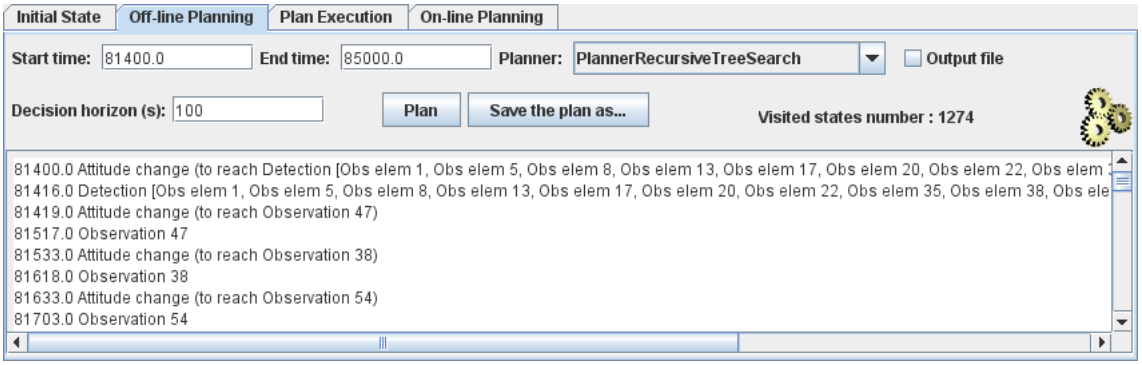


FIG. 6.11: Onglet *Off-line Planning* de l'environnement de simulation

5.4 Simulation de l'exécution d'un plan construit hors ligne

Une fois qu'un plan est construit sur la base du modèle bord, il est possible de l'exécuter dans le contexte réel du scénario de la mission d'observation défini. Pour cela, l'onglet *Plan Execution* permet à l'utilisateur de charger un plan précédemment enregistré au format XML (voir section 6), et de l'exécuter sur un certain horizon inclus dans l'horizon du scénario, avec un certain ratio entre le temps simulé et le temps réel. Comme le montre la figure 6.12, l'état réel courant du système, son état estimé à bord du satellite et l'action courante exécutée par le satellite sont alors visibles dans la partie inférieure de l'onglet.

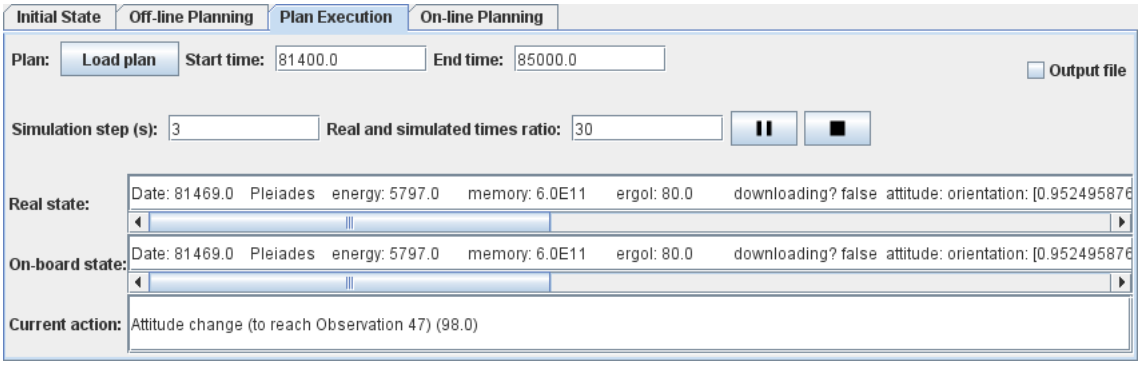


FIG. 6.12: Onglet *Plan Execution* de l'environnement de simulation

5.5 Exécution temps-réel et planification en ligne

Le quatrième et dernier onglet, intitulé *On-line Planning*, est sûrement le plus important des quatre. Il permet, une fois les modèles réel et bord chargés, de lancer l'exécution temps-réel du scénario entre une date de début et une date de fin, en spécifiant le planificateur bord utilisé par le satellite ainsi que la taille de l'horizon de décision associé.
En cours d'exécution, la partie inférieure gauche de l'onglet permet à l'utilisateur de suivre en temps-réel, l'évolution de l'état réel de l'environnement, de l'état estimé à bord du satellite, de l'action courante exécutée par le satellite, de la valeur de la prochaine échéance, et du gain, c'est-à-dire l'évaluation de l'état des objectifs de la mission.
La partie droite de l'onglet permet, quant à elle, de visualiser en temps-réel les délibérations de la tâche délibérative du logiciel de vol, c'est-à-dire le meilleur plan courant construit par le planificateur bord dans l'état courant.

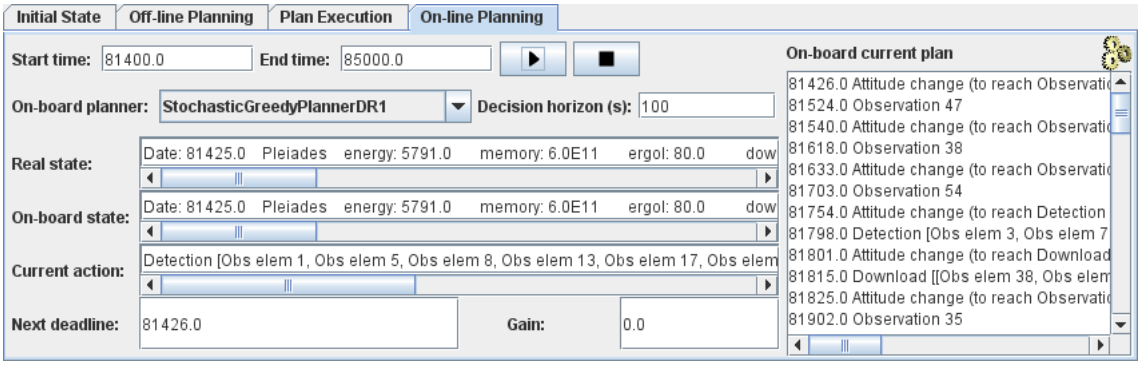


FIG. 6.13: Onglet *On-line Planning* de l'environnement de simulation

6 Entrées / sorties

6.1 Modèle

Ce que l'on appelle "modèle" est la réunion de l'horizon temporel du scénario, du modèle du satellite d'observation (orbite et manœuvres orbitales), des modèles des différents centres de missions, et du modèle des objectifs de la mission. Qu'il soit réel (c'est-à-dire simulant l'évolution réelle du système) ou bord (c'est-à-dire utilisé pour raisonner à bord du satellite), un modèle peut être sauvegardé sous la forme d'un fichier au format XML. La figure 6.14 représente un fichier XML décrivant un modèle réel composé d'un satellite, de deux centres de mission, de deux requêtes d'observation et d'une manœuvre orbitale sur un scénario d'une vingtaine d'heures.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- Modelise les differents elements du systeme et leurs
      evolutions au cours du scenario -->
<model>
  <!-- Date de debut du scenario -->
  <scenarioStartTime>0.0</scenarioStartTime>
  <!-- Date de fin du scenario -->
  <scenarioEndTime>82500.0</scenarioEndTime>
  <!-- Description du satellite -->
  <satellite name="Spot">
    <!-- Description de l orbite du satellite -->
    <orbit>
      <t0 definition="date des elements orbitaux">
           2.186219149632E8</t0>
      <i0 definition="inclinaison">1.723044888104866</i0>
      <om0 definition="argument du perigee a t0">
           1.3472475748824548</om0>
      <e definition="excentricite">1.025E-4</e>
      <Om0 definition="ascension droite du noeud
           ascendant a t0">0.9106953503981212</Om0>
      <m0 definition="anomalie moyenne a t0">
           1.5764354856080902</m0>
      <n0 definition="mouvement moyen">
           0.0010326782211446627</n0>
    </orbit>
    <!-- Un ensemble de manoeuvres orbitales -->
    <orbitalManoeuvresSet>
      <manoeuvre index="0">
        <startTime>34679.0</startTime>
        <duration>32.0</duration>
        <emissionTime>3042.0</emissionTime>
        <status>nonRealisee</status>
        <startAttitude>
          <position xx="-0.21" xy="-0.21" xz="0.95" yx="
               0.45" yy="-0.88" yz="-0.09" zx="0.86" zy="
               0.41" zz="0.28"/>
          <speed x="0.05" y="0.02" z="0.0"/>
        </startAttitude>
        <endAttitude>
          <position xx="-0.08" xy="0.12" xz="-0.98" yx="
               0.3" yy="0.94" yz="0.09" zx="0.94" zy="
               -0.29" zz="-0.11"/>
          <speed x="0.03" y="0.03" z="0.0"/>
        </endAttitude>
      </manoeuvre>
    </orbitalManoeuvresSet>
  </satellite>
  <!-- Liste des centres de mission utilises dans le
       scenario -->
  <missionCentersSet>
    <missionCenter name="Centre de Kourou">
      <station name="Centre de Kourou">
        <longitude>-0.9075712110370514</longitude>
        <latitude>0.08726646259971647</latitude>
        <altitude>0.0</altitude>
        <cap>0.0</cap>
      </station>
    </missionCenter>
    <missionCenter name="Centre de Toulouse">
      <station name="Centre de Toulouse">
        <longitude>0.02356194490192345</longitude>
        <latitude>0.7696902001294994</latitude>
        <altitude>0.0</altitude>
        <cap>0.0</cap>
      </station>
    </missionCenter>
  </missionCentersSet>
  <!-- Liste des objectifs de la mission -->
  <goals>
    <complexObservation index="1">
      <!-- Demandeur de l observation -->
      <user>CNES</user>
      <!-- Date d emission de la requete par un centre de
           mission -->
      <emissionTime>0.0</emissionTime>
      <!-- Priorite associee a l observation complexe -->
      <priority>1</priority>
      <!-- Date de teledechargement au plus tard de l
           image -->
      <downloadDeadline>82500.0</downloadDeadline>
      <!-- Statut initial de l observation complexe -->
      <status>nonCommencee</status>
      <!-- Liste des observations elementaires composant
           l observation complexe -->
      <basicObservation index="1" name="Obs 1">
        <!-- Observation stereoscopique associee si elle
             existe -->
        <twinBasicObservationIndex>0</
             twinBasicObservationIndex>
        <!-- Duree de l observation -->
        <duration>10.0</duration>
        <!-- Vitesse de balayage -->
        <scanningSpeed>1.0</scanningSpeed>
        <!-- Statut initial de l observation elementaire
             -->
        <status>nonRealisee</status>
        <!-- Place memoire necessaire pour memoriser l
             image a bord -->
        <memoryConsumption>1.0E9</memoryConsumption>
        <!-- Mode d observation (normal ou stereoscopique
             ) -->
        <mode>mode 1</mode>
        <!-- Date de la derniere detection de nuages au
             dessus de la zone a observer -->
        <lastDetectionTime>0.0</lastDetectionTime>
        <!-- Date de realisation -->
        <realizationTime>0.0</realizationTime>
        <!-- La zone geographique a observer -->
        <zone name="Zone 1">
          <longitude>-0.07534870026839528</longitude>
          <latitude>0.5181487325949439</latitude>
          <altitude>0.0</altitude>
          <!-- Azimuth d observation de la zone -->
          <cap>0.5227007575230708</cap>
          <dimension>40000.0</dimension>
          <!-- Liste initiale des previsions de
               couverture nuageuse au dessus de la zone a
               chaque survol (ici un seul survol) -->
          <cloudCover>
            <echantillon time="82110.0" value="0.25"/>
          </cloudCover>
        </zone>
      </basicObservation>
    </complexObservation>
    <complexObservation index="2">
      <user>CNES</user>
      <emissionTime>50200.0</emissionTime>
      <priority>2</priority>
      <downloadDeadline>82500.0</downloadDeadline>
      <status>nonCommencee</status>
      <basicObservation index="2" name="Obs 2">
        <twinBasicObservationIndex>0</
             twinBasicObservationIndex>
        <duration>10.0</duration>
        <scanningSpeed>1.0</scanningSpeed>
        <status>nonRealisee</status>
        <memoryConsumption>1.0E9</memoryConsumption>
        <mode>mode 1</mode>
        <lastDetectionTime>0.0</lastDetectionTime>
        <realizationTime>0.0</realizationTime>
        <zone name="Zone 2">
          <longitude>-0.04776690584405534</longitude>
          <latitude>0.43728099038451984</latitude>
          <altitude>0.0</altitude>
          <cap>0.0</cap>
          <dimension>20000.0</dimension>
          <cloudCover>
            <echantillon time="82110.0" value="0.39"/>
          </cloudCover>
        </zone>
      </basicObservation>
    </complexObservation>
  </goals>
</model>
```

FIG. 6.14: Exemple de fichier XML d'un modèle réel

6.2 Plan

Les plans construits hors ligne peuvent être sauvegardés sous la forme de fichiers XML, puis chargés pour être simulés. Un plan sera décrit par un état de départ du système, et une séquence d'actions. Un aperçu du fichier de sauvegarde d'un plan d'activités du satellite est représenté sur la figure 6.16.

6.3 Exécution

Lors d'une simulation temps-réel, un fichier de sortie au format CSV est créé. Il décrit à chaque boucle réactive du logiciel de vol[4], l'état réel du système, l'état bord correspondant à l'état réel tel qu'il est perçu par le satellite, les délibérations émises par la tâche délibérative, les actions engagées par la tâche réactive et les arrivées de nouvelles requêtes au cours de cette boucle de réaction. La figure 6.15 représente un échantillon d'un fichier de sortie créé au cours d'une simulation temps-réel.

Reaction #	Real state				On-board state				Value	Deliberation	Commitment	New requests
	Date	Energy	Memory	Ergol	Date	Energy	Memory	Ergol				
0	82110	5800	6E+11	80	82110	5800	6E+11	80	0			
1	82111	5800	6E+11	80	82111	5800	6E+11	80	0			
2	82112	5800	6E+11	80	82112	5800	6E+11	80	0			
3	82113	5800	6E+11	80	82113	5800	6E+11	80	0			
4	82114	5800	6E+11	80	82114	5800	6E+11	80	0			
5	82115	5800	6E+11	80	82115	5800	6E+11	80	0			
6	82116	5800	6E+11	80	82116	5800	6E+11	80	0			
7	82117	5800	6E+11	80	82117	5800	6E+11	80	0		Attitude change (to reach Detection [Obs elem 1, Obs elem 2])	
8	82118	5800	6E+11	80	82118	5800	6E+11	80	0			
9	82119	5800	6E+11	80	82119	5800	6E+11	80	0			
10	82120	5800	6E+11	80	82120	5800	6E+11	80	0			Obs. comp. 3
11	82121	5800	6E+11	80	82121	5800	6E+11	80	0	Detection [Obs elem 1, Obs elem 2]		
12	82122	5800	6E+11	80	82122	5800	6E+11	80	0	Detection [Obs elem 1, Obs elem 2, Obs elem 3]		
13	82123	5800	6E+11	80	82123	5800	6E+11	80	0	Detection [Obs elem 1, Obs elem 2, Obs elem 3]		
14	82124	5800	6E+11	80	82124	5800	6E+11	80	0	Detection [Obs elem 1, Obs elem 2, Obs elem 3]		
15	82125	5800	6E+11	80	82125	5800	6E+11	80	0			
16	82126	5800	6E+11	80	82126	5800	6E+11	80	0			
17	82127	5800	6E+11	80	82127	5800	6E+11	80	0		Detection [Obs elem 1, Obs elem 2, Obs elem 3]	
18	82128	5800	6E+11	80	82128	5800	6E+11	80	0	Attitude change (to reach Observation 2)		
19	82129	5800	6E+11	80	82129	5800	6E+11	80	0			
20	82130	5800	6E+11	80	82130	5800	6E+11	80	0	Attitude change (to reach Observation 2)		
21	82131	5800	6E+11	80	82131	5800	6E+11	80	0			
22	82132	5800	6E+11	80	82132	5800	6E+11	80	0			
23	82133	5800	6E+11	80	82133	5800	6E+11	80	0			
24	82134	5799,22	6E+11	80	82134	5799,22	6E+11	80	0	Attitude change (to reach Observation 1)		
25	82135	5798,34	6E+11	80	82135	5798,34	6E+11	80	0			
26	82136	5797,46	6E+11	80	82136	5797,46	6E+11	80	0			
27	82137	5797,69	6E+11	80	82137	5797,69	6E+11	80	0		Attitude change (to reach Observation 1)	
28	82138	5797,79	6E+11	80	82138	5797,79	6E+11	80	0	Observation 1		
29	82139	5797,89	6E+11	80	82139	5797,89	6E+11	80	0			
30	82140	5797,98	6E+11	80	82140	5797,98	6E+11	80	0	Observation 1		
31	82141	5798,05	6E+11	80	82141	5798,05	6E+11	80	0			
32	82142	5798,1	6E+11	80	82142	5798,1	6E+11	80	0			
33	82143	5798,13	6E+11	80	82143	5798,13	6E+11	80	0			
34	82144	5798,14	6E+11	80	82144	5798,14	6E+11	80	0			
35	82145	5798,14	6E+11	80	82145	5798,14	6E+11	80	0		Observation 1	
36	82146	5798,14	6E+11	80	82146	5798,14	6E+11	80	0,16	Attitude change (to reach Observation 2)		
37	82147	5798,14	6E+11	80	82147	5798,14	6E+11	80	0,16	Attitude change (to reach Observation 2)		
38	82148	5798,14	6E+11	80	82148	5798,14	6E+11	80	0,16			
39	82149	5798,14	6E+11	80	82149	5798,14	6E+11	80	0,16			
40	82150	5798,14	6E+11	80	82150	5798,14	6E+11	80	0,16	Attitude change (to reach Observation 2)		
41	82151	5798,14	6E+11	80	82151	5798,14	6E+11	80	0,16		Attitude change (to reach Observation 2)	
42	82152	5798,14	6E+11	80	82152	5798,14	6E+11	80	0,16	Observation 2		
43	82153	5798,14	6E+11	80	82153	5798,14	6E+11	80	0,16			
44	82154	5798,14	6E+11	80	82154	5798,14	6E+11	80	0,16			
45	82155	5798,14	6E+11	80	82155	5798,14	6E+11	80	0,16	Observation 2		
46	82156	5798,14	6E+11	80	82156	5798,14	6E+11	80	0,16	Observation 2		
47	82157	5798,14	6E+11	80	82157	5798,14	6E+11	80	0,16			
48	82158	5798,14	6E+11	80	82158	5798,14	6E+11	80	0,16			
49	82159	5798,14	6E+11	80	82159	5798,14	6E+11	80	0,16		Observation 2	
50	82160	5798,14	6E+11	80	82160	5798,14	6E+11	80	0,5			
51	82161	5798,14	6E+11	80	82161	5798,14	6E+11	80	0,5			

FIG. 6.15: Exemple de fichier de sortie d'une exécution temps-réel

[4]La boucle réactive du logiciel de vol implémenté a une période d'une seconde.

```
<?xml version="1.0" encoding="UTF-8"?>
<plan>
  <!-- Etat de depart -->
  <startState>
    <time>82110.0</time>
    <satelliteState name="Spot" energyLevel="5800.0" memoryLevel="6.0E11" ergolLevel="80.0" isDownloading="false">
      <manoeuvreStateList></manoeuvreStateList>
      <attitude>
        <position xx="-0.25" xy="0.96" xz="-0.10" yx="-0.54" yy="-0.23" yz="-0.80" zx="-0.79" zy="-0.15" zz="0.58"/>
        <speed x="9.91E-4" y="-2.4E-4" z="-1.56E-4"/>
      </attitude>
      <orbitalPosition x="-736360.21" y="-5789137.74" z="4224033.23"/>
      <translationSpeed x="-1922.72" y="-4075.58" z="-5921.21"/>
    </satelliteState>
    <goalState>
      <complexObservationState index="1" status="nonCommencee">
        <basicObservationState index="1" status="nonRealisee" memoryConsumption="1.0E9" downloadDuration="2.0"
            cloudCover="0.0" lastDetectionTime="0.0" realizationTime="0.0" quality="0.0"/>
      </complexObservationState>
      <complexObservationState index="2" status="nonCommencee">
        <basicObservationState index="2" status="nonRealisee" memoryConsumption="1.0E9" downloadDuration="2.0"
            cloudCover="0.0" lastDetectionTime="0.0" realizationTime="0.0" quality="0.0"/>
      </complexObservationState>
    </goalState>
  </startState>
  <!-- Liste des actions successives -->
  <actionsList>
    <action name="Attitude change (to reach Detection [Obs elem 1, Obs elem 2])" class="class simulation.AttitudeChange
        " duration="16.0" energyConsumption="0.0" memoryConsumption="0.0" ergolConsumption="0.0" energyProduction="
        8.37" memoryProduction="0.0">
      <startState>
        <time>82110.0</time>
        <satelliteState name="Spot" energyLevel="5800.0" memoryLevel="6.0E11" ergolLevel="80.0" isDownloading="false">
          <manoeuvreStateList></manoeuvreStateList>
          <attitude>
            <position xx="-0.25" xy="0.96" xz="-0.1" yx="-0.54" yy="-0.23" yz="-0.8" zx="-0.79" zy="-0.15" zz="0.58"/>
            <speed x="9.91E-4" y="-2.4E-4" z="-1.56E-4"/>
          </attitude>
          <orbitalPosition x="-736360.21" y="-5789137.74" z="4224033.23"/>
          <translationSpeed x="-1922.72" y="-4075.58" z="-5921.21"/>
        </satelliteState>
        <goalState>
          <complexObservationState index="1" status="nonCommencee">
            <basicObservationState index="1" status="nonRealisee" memoryConsumption="1.0E9" downloadDuration="2.0"
                cloudCover="0.0" lastDetectionTime="0.0" realizationTime="0.0" quality="0.0"/>
          </complexObservationState>
          <complexObservationState index="2" status="nonCommencee">
            <basicObservationState index="2" status="nonRealisee" memoryConsumption="1.0E9" downloadDuration="2.0"
                cloudCover="0.0" lastDetectionTime="0.0" realizationTime="0.0" quality="0.0"/>
          </complexObservationState>
        </goalState>
      </startState>
      <endAttitude>
        <position xx="-0.27" xy="0.96" xz="0.03" yx="-0.86" yy="-0.23" yz="-0.43" zx="-0.41" zy="-0.15" zz="0.89"/>
        <speed x="9.91E-4" y="-2.4E-4" z="-1.56E-4"/>
      </endAttitude>
    </action>
    <action name="Detection [Obs elem 1, Obs elem 2]" class="class simulation.Detection" duration="3.0"
        energyConsumption="3.0" memoryConsumption="0.0" ergolConsumption="0.0" energyProduction="1.57"
        memoryProduction="0.0">
      <startState>
        <time>82126.0</time>
        <satelliteState name="Spot" energyLevel="5800.0" memoryLevel="6.0E11" ergolLevel="80.0" isDownloading="false">
          <manoeuvreStateList></manoeuvreStateList>
          <attitude>
            <position xx="-0.27" xy="0.96" xz="0.03" yx="-0.86" yy="-0.23" yz="-0.43" zx="-0.41" zy="-0.15" zz="0.89"/>
            <speed x="9.91E-4" y="-2.4E-4" z="-1.56E-4"/>
          </attitude>
          <orbitalPosition x="-766986.05" y="-5853520.27" z="4128775.38"/>
          <translationSpeed x="-1909.89" y="-3976.28" z="-5992.45"/>
        </satelliteState>
        <goalState>
          <complexObservationState index="1" status="nonCommencee">
            <basicObservationState index="1" status="nonRealisee" memoryConsumption="1.0E9" downloadDuration="2.0"
                cloudCover="0.0" lastDetectionTime="0.0" realizationTime="0.0" quality="0.0"/>
          </complexObservationState>
          <complexObservationState index="2" status="nonCommencee">
            <basicObservationState index="2" status="nonRealisee" memoryConsumption="1.0E9" downloadDuration="2.0"
                cloudCover="0.0" lastDetectionTime="0.0" realizationTime="0.0" quality="0.0"/>
          </complexObservationState>
        </goalState>
      </startState>
    </action>
  </actionsList>
</plan>
```

FIG. 6.16: Exemple de fichier XML d'un plan

Chapitre 7

Évaluation des mécanismes de décision mis en place

1 Évaluation hors ligne des mécanismes de décision

1.1 Influence des différents paramètres

Qu'ils soient réactif ou délibératif, les mécanismes de décision décrits jusqu'à maintenant sont fortement dépendants d'un certain nombre de paramètres comme la taille de l'horizon de décision, la taille de l'horizon de planification (définis dans le chapitre 5), ou les valeurs des probabilités p, q et r utilisées dans la règle de décision 1 bruitée, SDR1.

1.1.1 Influence de la taille de l'horizon de décision

Dans cette section nous déterminons la taille optimale de l'horizon de décision à utiliser dans les mécanismes de décision à bord du satellite.
Comme l'algorithme de planification est aussi basé sur une règle de décision (DR1), nous allons déterminer cette taille optimale pour les deux règles de décision, DR1 et DR2, présentées au chapitre 5.

Pour cela, nous avons utilisé un scénario de référence d'une durée d'une heure (environ une demi-révolution éclairée) et comportant 33 requêtes d'observation. La figure 7.1 représente la qualité du plan construit par la règle de décision DR1 et le temps qui lui a été nécessaire pour calculer ce plan, en fonction de la taille de l'horizon de décision variant entre zéro seconde et une heure.

La figure 7.2 représente ces mêmes grandeurs dans le cas de la règle de décision DR2.

Choix de la taille de l'horizon de décision Du point de vue de la qualité des plans construits, on remarque qu'un maximum est atteint pour des horizons de décision dont la taille avoisine les 100 secondes, aussi bien pour DR1 que pour DR2. Cette valeur

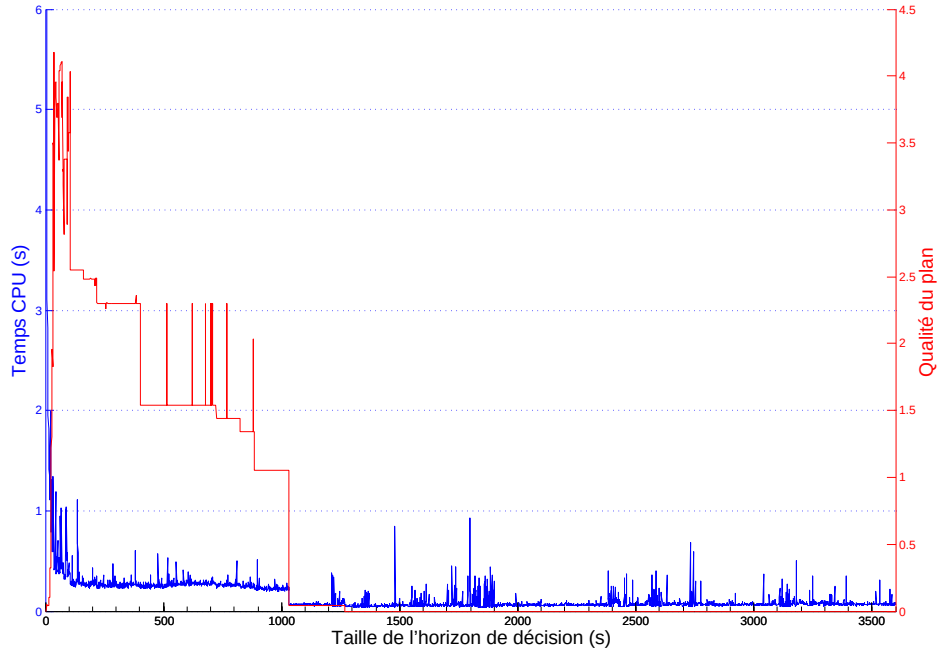


FIG. 7.1: Performances de DR1 en fonction de la taille de l'horizon de décision

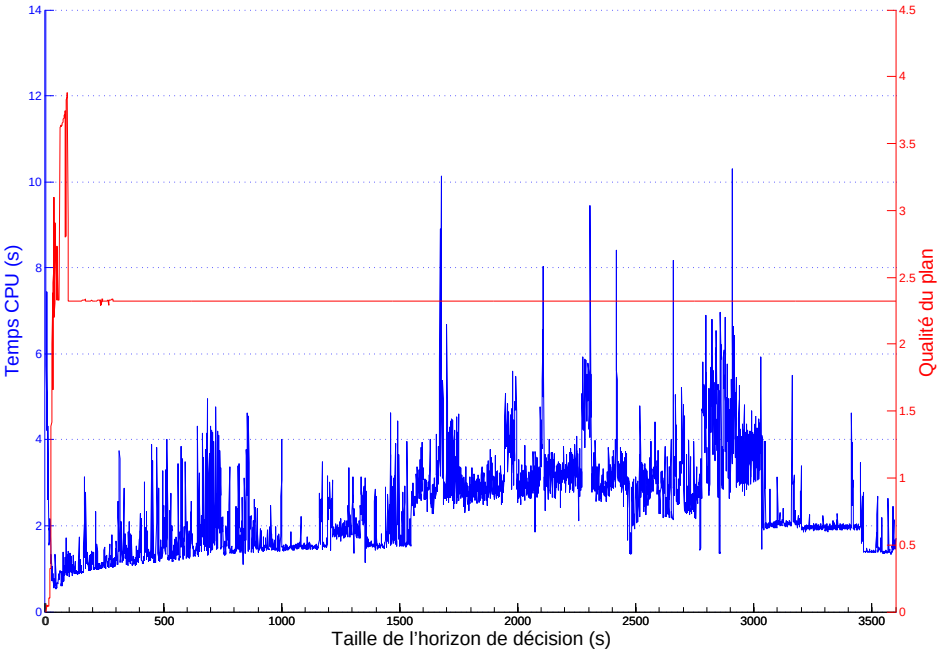


FIG. 7.2: Performances de DR2 en fonction de la taille de l'horizon de décision

correspond à un horizon de deux à trois actions en avant de l'état de décision. La chute brutale de la valeur des plans construits par DR1 pour un horizon de décision supérieur à 100 secondes s'explique par la nature même de DR1 dont le principe est de choisir l'action la plus prioritaire, même si elle est très loin de l'état de décision, et que des actions moins prioritaires peuvent être réalisées avant.

Le palier présent sur la figure 7.2 pour un horizon de décision supérieur à 120 secondes s'explique quant à lui par la nature de DR2 qui choisit toujours l'action la plus proche de l'instant de décision : passées 120 secondes, l'action la plus proche est toujours la même, et les plans ainsi construits sont identiques.

Lors de l'implémentation du logiciel de vol, la valeur de l'horizon de décision des règles de décision utilisées a donc été fixée à 100 secondes.

Choix de la règle de décision Sur le scénario de référence, les qualités optimales des plans construits par les deux règles de décision sont très proches : autour de 4. Le choix de l'implémentation de l'une ou l'autre règle à bord du satellite s'est donc fait principalement sur le temps de calcul nécessaire à la construction d'un plan. Pour vérifier l'hypothèse synchrone[1], la règle de décision implémentée doit être capable de prendre une décision en un laps de temps inférieur à la période de la boucle réactive du logiciel de vol, qui est d'une seconde. Or le mécanisme de décision utilisé par DR1 est plus adapté à une décision rapide car il détermine les différentes actions envisageables, par niveaux de priorité décroissants (lorsqu'une action a été trouvée, les niveaux de priorité inférieurs ne sont pas traités), alors que celui de DR2 nécessite de déterminer à chaque prise de décision l'ensemble complet des actions envisageables sur l'horizon de décision. Ainsi, on remarque qu'il est fréquent (voir très fréquent pour des horizons de décision grands) que le temps de calcul nécessaire à DR2 pour prendre une décision, dépasse la seconde.

C'est pourquoi nous avons choisi d'implémenter la règle de décision DR1 dans la partie réactive du logiciel de vol.

1.1.2 Réglage des probabilités utilisées dans l'algorithme d'optimisation

L'algorithme d'optimisation implémenté dans la tâche délibérative du logiciel de vol est un algorithme glouton stochastique itéré défini à la section 3.1.1 du chapitre 5. Il fait intervenir trois paramètres (les trois probabilités p, q, r définies sur la figure 5.5) à valeur dans l'intervalle $[0; 1]$. Avant d'utiliser cet algorithme en ligne, dans le contexte de la mission d'observation de la Terre, les valeurs optimales de ces trois paramètres ont du être déterminées hors ligne.

[1]L'hypothèse synchrone consiste à supposer l'existence d'un temps global, et que les réactions du système sont instantanées. Ainsi deux réactions du système exécutées à la même date seront considérées comme simultanées.

Réglage du paramètre r À priorités égales, les deux propriétés qui caractérisent le mieux la valeur d'une requête d'observation sont sa date de début au plus tôt et la couverture nuageuse au dessus de la zone à observer. Cependant, vaut-il mieux observer rapidement une zone nuageuse ou observer plus tardivement une zone très peu nuageuse ? La réponse à cette question dépend de la valeur de l'observation élémentaire en question au moment de la prise de décision. C'est pourquoi nous n'avons fait aucun choix a priori, et avons fixé la valeur du paramètre r à 0.5 : au cours d'une descente de l'algorithme de recherche, lorsqu'une action d'observation est décidée, la zone à observer choisie sera celle débutant au plus tôt, avec une probabilité de 0.5 et celle de plus faible couverture nuageuse, avec une probabilité de 0.5 (sachant que seule la solution qui aboutira au plan de meilleure qualité sera finalement retenue).

Réglages des paramètres p **et** q Les paramètres p et q ont été réglés expérimentalement : nous avons fait varier ces deux paramètres entre les valeurs 0.5 et 1 avec un pas de 0.1, et pour chaque couple de valeurs (p, q) nous avons lancé vingt exécutions de dix minutes de l'algorithme glouton stochastique itéré sur un même scénario d'une heure.

La figure 7.3 représente, à q fixé, la qualité des différentes délibérations rendues par l'algorithme au cours du temps, pour des valeurs de p dans l'ensemble $\{0.5; 0.6; 0.7; 0.8; 0.9; 1\}$. De la même façon, la figure 7.4 représente, à p fixé, la qualité des différentes délibérations rendues par l'algorithme au cours du temps, pour des valeurs de q dans l'ensemble $\{0.5; 0.6; 0.7; 0.8; 0.9; 1\}$.

Les meilleurs résultats sont obtenus pour les valeurs de p et q suivantes (solutions encadrées en rouge sur les figures 7.3 et 7.4) :

$$\begin{aligned} p &= 0.9 \\ q &= 0.8 \end{aligned}$$

Ces deux probabilités étant proches de 1, l'algorithme d'optimisation est donc très guidé par l'heuristique (DR1).

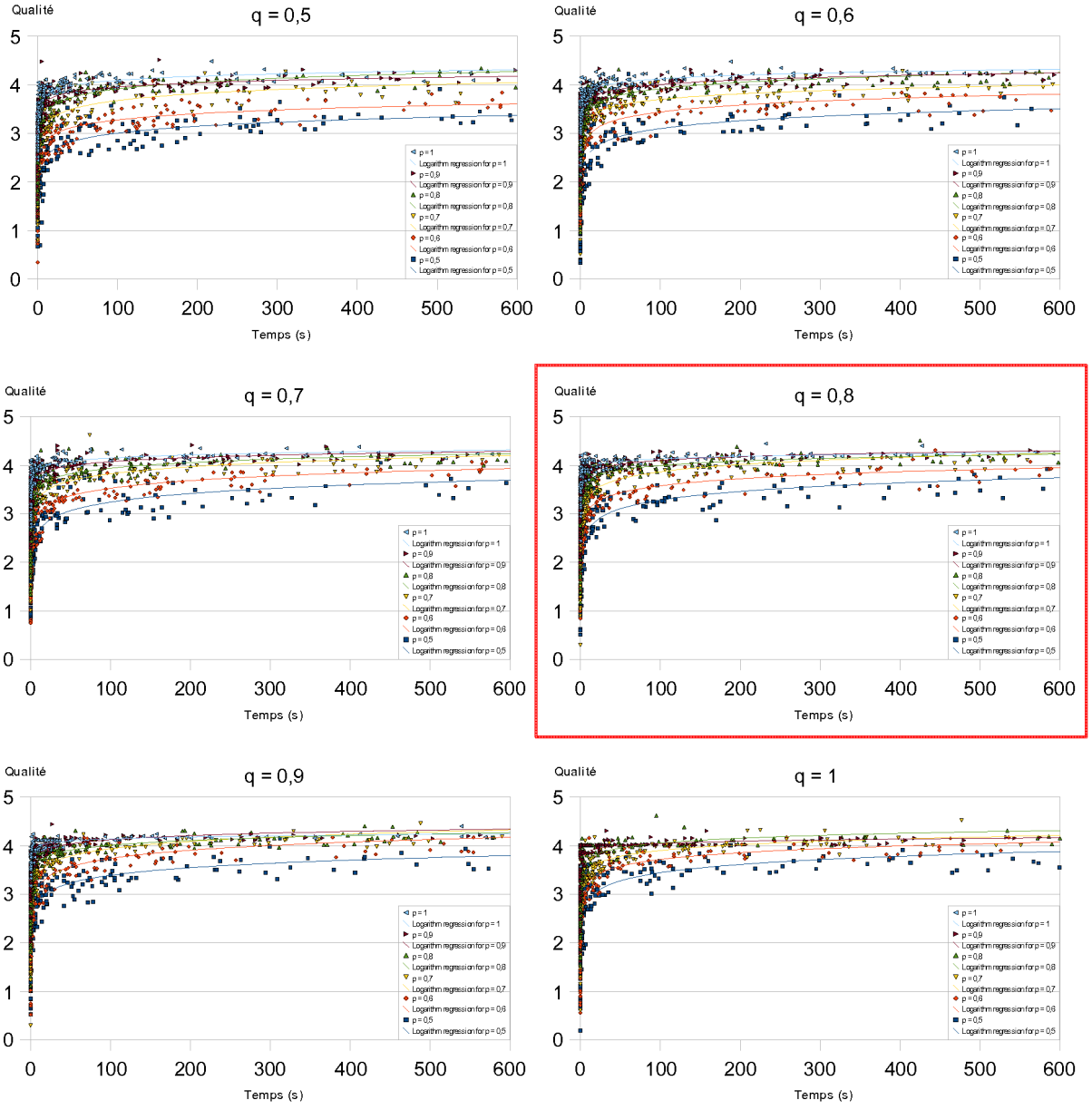


FIG. 7.3: Réglage du paramètre p

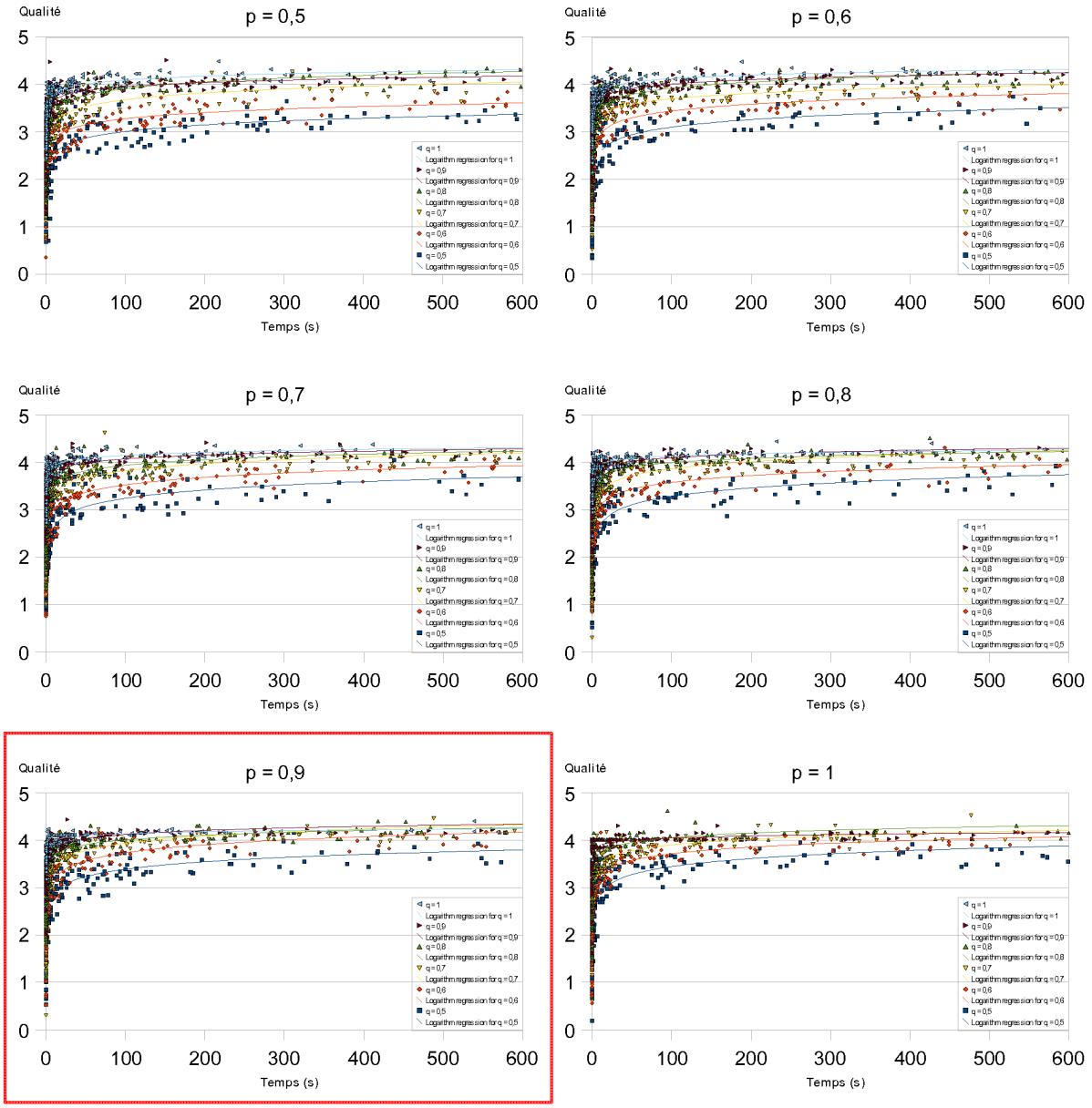


FIG. 7.4: Réglage du paramètre q

1.2 Évaluation des performances de l'algorithme d'optimisation

1.2.1 Vitesse de raisonnement

Deux paramètres caractérisent la vitesse de raisonnement de la tâche délibérative : le temps nécessaire à l'algorithme pour rendre une première délibération et la durée moyenne

d'une descente dans l'arbre de recherche. Comme cela a été vu au chapitre 5, le premier paramètre correspond au temps de réponse de la règle de décision utilisée dans le logiciel de vol (la première délibération rendue par la tâche délibérative est la décision que rendrait la règle de décision). Il est donc supposé instantané, c'est-à-dire largement inférieur à la période de la boucle réactive du logiciel de vol, qui est d'une seconde.

Le deuxième paramètre est quant à lui fortement dépendant de la taille du problème à résoudre par la tâche délibérative, et en particulier du scénario, de l'horizon de décision et de l'horizon de planification de l'algorithme d'optimisation.

Pour un scénario de 100 minutes (soit environ une révolution) avec 60 requêtes d'observation, le tableau 7.1 représente les caractéristiques temporelles des 9549 descentes effectuées pendant une heure par un algorithme glouton stochastique itéré dont l'horizon de planification a été fixé à 100 minutes (durée moyenne de l'horizon de planification au cours de la mission) et l'horizon de décision à 100 secondes.

Durée maximale d'une descente (ms)	3949
Durée minimale d'une descente (ms)	243
Durée moyenne d'une descente (ms)	374.29

TAB. 7.1: Durée d'une descente de l'algorithme d'optimisation

On remarque que les descentes dans l'arbre de recherche sont très rapides (quelques centaines de millisecondes en moyenne), mais on ne peut pas garantir de borne supérieure de leurs durées (ici, la durée de la plus longue descente atteint presque les quatre secondes).

1.2.2 Qualité des délibérations

Afin d'évaluer la qualité des délibérations rendues (c'est-à-dire des améliorations successives du meilleur plan construit) par la tâche délibérative du logiciel de vol, nous avons étudié, sur un scénario d'une heure (30 requêtes d'observation) l'évolution de la qualité du meilleur plan construit par l'algorithme d'optimisation, en fonction du temps de raisonnement, et nous l'avons comparée à la qualité du plan construit sur le même scénario par la règle de décision DR1. La figure 7.5 représente cette évolution au cours du temps. Chaque point bleu correspond à la valeur de la qualité d'un plan construit par la tâche délibérative et qui a amélioré la valeur de la qualité du meilleur plan produit jusqu'alors. On remarque qu'au delà de dix secondes de raisonnement, en moyenne, la tâche délibérative (représentée par la ligne rouge continue sur la figure 7.5) réussit à construire des plans de meilleure qualité que ne le ferait la règle de décision DR1 (représentée par la ligne verte discontinue sur la figure 7.5).

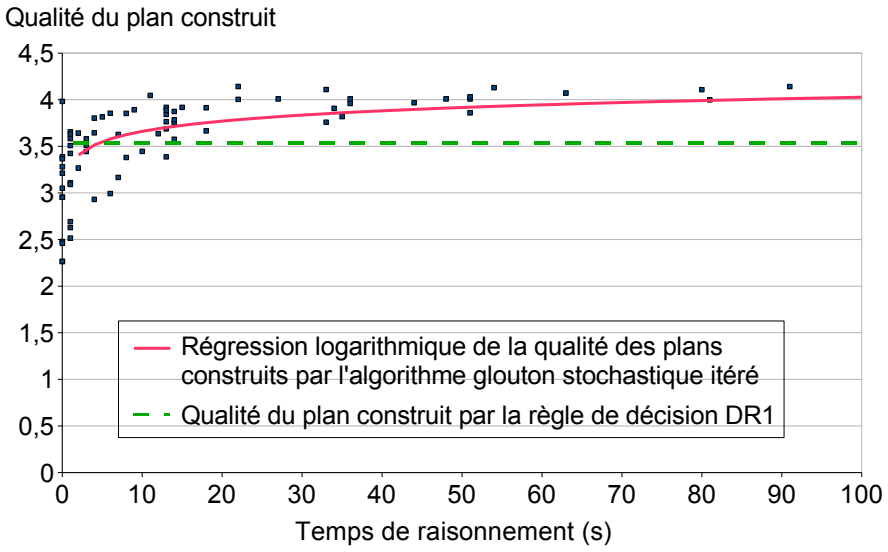


FIG. 7.5: Comparaison hors ligne entre l'algorithme d'optimisation et la règle de décision DR1

2 Évaluation en ligne des mécanismes de décision

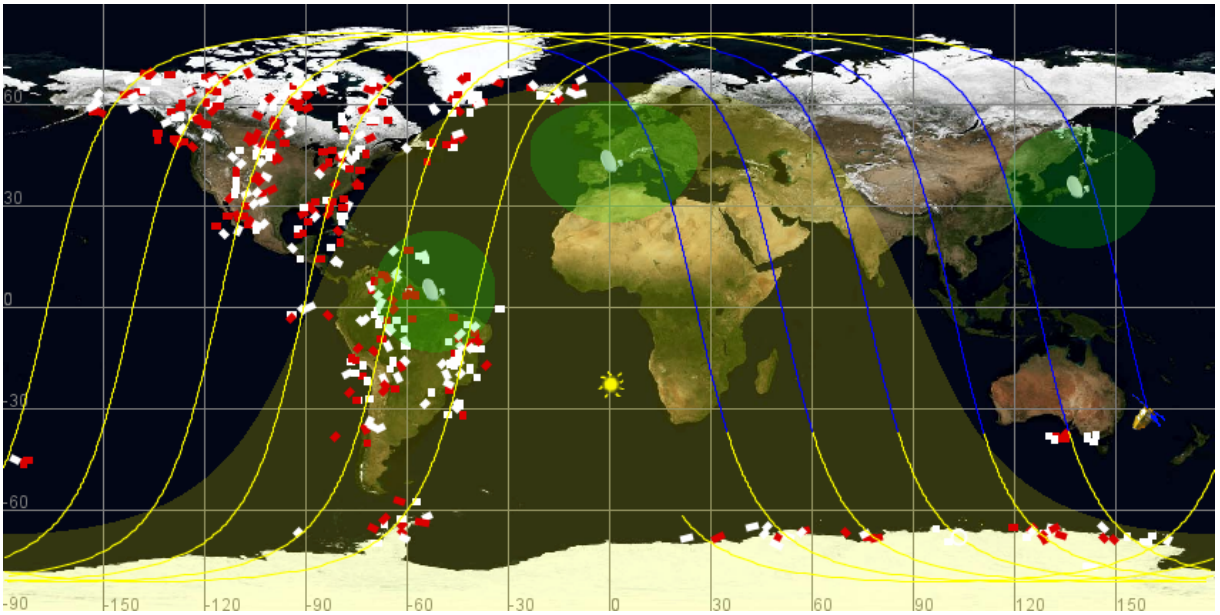


FIG. 7.6: Scénario de référence de 10 heures avec 3 centres de mission et 400 requêtes d'observation

2.1 Comportement du logiciel de vol

Des simulations temps-réel sur plusieurs heures nous ont permis, dans un premier temps, de valider le comportement du satellite au cours de sa mission.

La figure 7.7 représente l'évolution des ressources à bord du satellite, pour un scénario test de dix heures, avec trois centres de mission (Kourou, Toulouse et Tokyo) et 400 requêtes d'observation dont environ la moitié sera envoyée au satellite en cours de mission (voir figure 7.6). Le niveau d'énergie du satellite décroît en période d'éclipse et croît en période d'éclairement. Son niveau de mémoire disponible décroît lorsque le satellite mémorise des données d'observation, et croît aux cours des actions de télédéchargement.

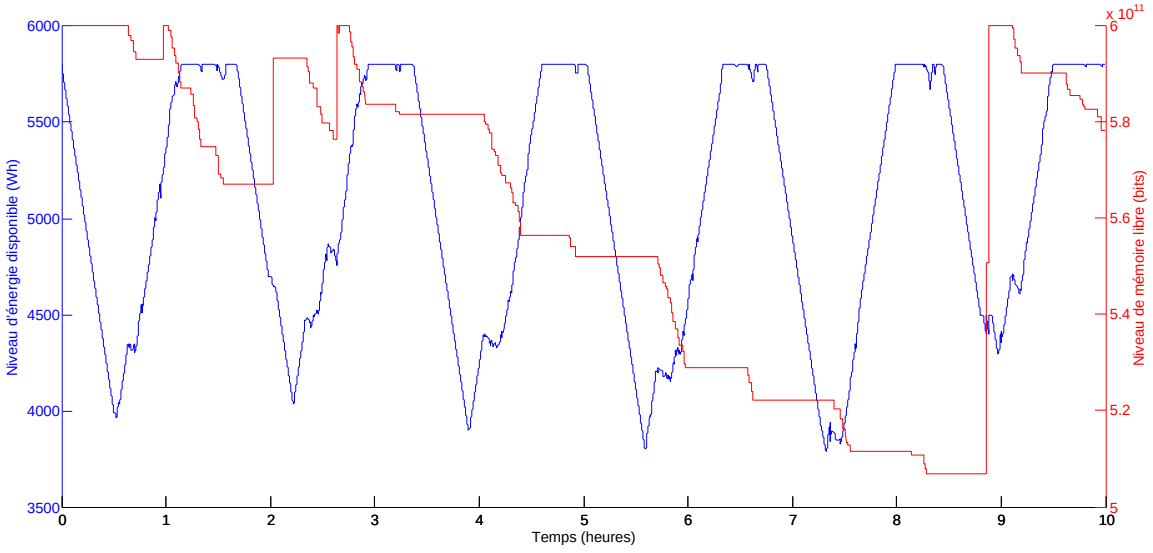


FIG. 7.7: Évolution des ressources disponibles à bord du satellite sur un scénario de 10 heures. Sur les 400 requêtes d'observation, 30% ont été réalisés dont 86% ont été télédéchargés avant la fin de l'horizon de simulation

La figure 7.8 quant à elle, représente le temps passé par le satellite dans chaque type d'action.

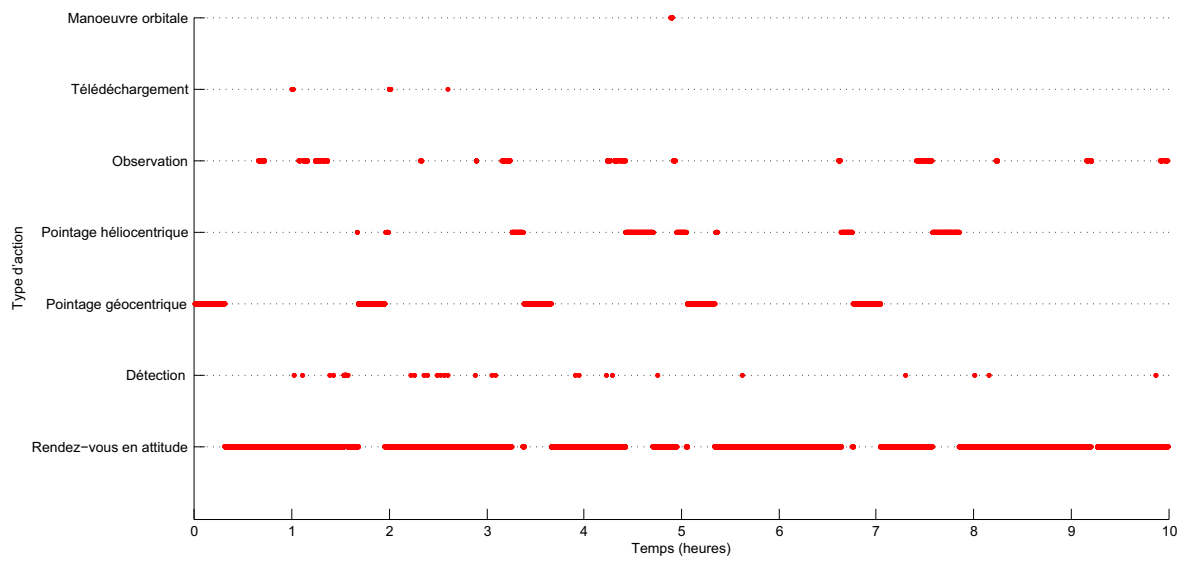


FIG. 7.8: Chronogramme des différentes activités du satellite sur un scénario de 10 heures et 400 requêtes d'observation

2.2 Intérêt de la planification à bord et en ligne

2.2.1 Planification sol - planification bord

Dans cette partie on cherche à montrer l'intérêt d'utiliser des mécanismes de décision à bord du satellite par rapport à ce qui est actuellement fait pour les satellites d'observation de type Spot, à savoir construire au sol un plan d'activités journalier et le transmettre quotidiennement au satellite qui se contente de l'exécuter tel quel.

Pour cela nous avons comparé deux simulations d'exécution de mission sur un scénario de référence d'une heure et de 60 requêtes d'observation. Pour la première simulation, le satellite est équipé des mécanismes de décision que nous avons développés (tâches réactive et délibérative) et la gestion de la mission est réalisée de manière autonome à bord du satellite. Pour la seconde simulation, nous avons reproduit la gestion de mission telle qu'elle est aujourd'hui réalisée : un plan est tout d'abord construit a priori, hors ligne, en laissant raisonner l'algorithme glouton stochastique itéré durant plusieurs heures sur un modèle de la mission, ne prenant en compte que des prévisions météorologiques sur les conditions d'ennuagement qui seront rencontrées.

Le tableau 7.2 montre les résultats obtenus au cours de ces deux simulations : une planification bord en ligne tenant compte des informations sur la détection de la couverture nuageuse permet de réaliser plus d'observations (trois de plus) et de sélectionner les zones qui sont réellement moins nuageuses, qu'une planification sol réalisée hors ligne.

	planification sol hors ligne	planification bord en ligne
nombre d'observations réalisées	12	15
nombre d'observations réalisées de priorité 1	1	3
nombre d'observations réalisées de priorité 2	2	3
nombre d'observations réalisées de priorité 3	9	9
nombre d'observations réalisées avec une couverture nuageuse $< 50\%$	6	12
nombre d'observations réalisées avec une couverture nuageuse $\geq 50\%$	6	3
couverture nuageuse moyenne de toutes les observations réalisées	50%	34%
qualité du plan d'activités exécuté	4.11	4.98

TAB. 7.2: Intérêt de la planification bord en ligne

2.2.2 Décision bord avec ou sans planification

Afin de montrer l'intérêt de l'utilisation à bord d'un algorithme d'optimisation, par rapport à la simple application d'une règle de décision, nous avons, sur un même scénario de référence d'une heure et de 60 requêtes d'observation, simulé le comportement temps-réel de deux satellites, l'un équipé d'une simple règle de décision (DR1) pour prendre des décisions à bord, et l'autre équipé en plus d'un mécanisme d'optimisation de ses décisions (une tâche délibérative) basé sur l'algorithme glouton stochastique décrit au chapitre 5. Le tableau 7.3 synthétise les résultats obtenus au cours des deux simulations. La tâche délibérative permet de réaliser plus d'observations (15 au lieu de 10), et de privilégier les zones les moins couvertes en nuages (34% de couverture nuageuse moyenne contre 48% sans tâche délibérative). Ces améliorations se font en particulier ressentir dans la qualité du plan d'activités réalisé par le satellite qui passe de 2.7 à 4.98.

	sans tâche délibérative	avec tâche délibérative
nombre d'observations réalisées	10	15
nombre d'observations réalisées de priorité 1	1	3
nombre d'observations réalisées de priorité 2	1	3
nombre d'observations réalisées de priorité 3	8	9
nombre d'observations réalisées avec une couverture nuageuse $< 50\%$	6	12
nombre d'observations réalisées avec une couverture nuageuse $\geq 50\%$	4	3
couverture nuageuse moyenne de toutes les observations réalisées	48%	34%
qualité du plan d'activités exécuté	2.7	4.98

TAB. 7.3: Simulation en ligne avec et sans tâche délibérative

2.3 Intérêt de la détection de la couverture nuageuse

Afin d'évaluer l'impact de la détection de la couverture nuageuse, à bord du satellite, nous avons rejoué le même scénario que précédemment avec deux satellites cette fois identiques du point de vue des mécanismes de décision (équipés du logiciel de vol complet : tâches réactive et délibérative), mais en interdisant au second de détecter la couverture nuageuse au dessus des zones à observer. Son estimation des couvertures nuageuses au dessus de chacune des zones à observer ne se fera que sur les prévisions météorologiques qui lui seront envoyées par les stations sol. Le tableau 7.4 représente l'état des objectifs à la fin des deux simulations. Dans le cas où le satellite détecte la couverture nuageuse, il réalise moins d'observations (15 au lieu de 18) notamment à cause du temps qu'il perd à réaliser des actions de détection de nuages. Cependant ces observations sont de bien meilleure qualité (la couverture nuageuse moyenne ne dépasse pas les 34% alors qu'elle atteint 55% si le satellite ne la détecte pas) ce qui augmente considérablement la qualité du plan d'activités exécuté par le satellite : d'une qualité de 2.32 sans détection, on passe à 4.98 avec détection.

	sans détection	avec détection
nombre d'observations réalisées	18	15
nombre d'observations réalisées de priorité 1	3	3
nombre d'observations réalisées de priorité 2	4	3
nombre d'observations réalisées de priorité 3	11	9
nombre d'observations réalisées avec une couverture nuageuse $< 50\%$	9	12
nombre d'observations réalisées avec une couverture nuageuse $\geq 50\%$	9	3
couverture nuageuse moyenne de toutes les observations réalisées	55%	34%
qualité du plan d'activités exécuté	2.32	4.98

TAB. 7.4: Simulation en ligne avec et sans détection de la couverture nuageuse à bord

---

La présentation de ces résultats a donné lieu à une communication internationale [Beaumet *et al.*, 2008b] au workshop "Scheduling and Planning Applications" de la conférence ICAPS'08 (International Conference on Automated Planning and Scheduling).

Conclusion et perspectives

Résumé

Au travers de ce manuscrit, nous avons tenté de montrer la possibilité et l'intérêt de déplacer les mécanismes de décision d'un satellite d'observation de la Terre, du sol vers le bord.

Nous avons ainsi adapté au cas d'un satellite agile, étendu et amélioré une architecture de contrôle autonome existante développée dans le cadre du projet AGATA. Cette architecture est centrée sur une tâche réactive capable de répondre instantanément aux stimuli de l'environnement, qui contrôle une tâche délibérative s'exécutant en parallèle et utilisant au mieux le temps dont elle dispose pour choisir de façon optimale la prochaine décision à prendre.

Un environnement de simulation temps-réel nous a permis de valider le comportement de cette architecture autonome, montrant qu'il est désormais réaliste de concevoir un satellite agile d'observation décidant de façon autonome et en temps-réel des actions à exécuter parmi des actions de manœuvre orbitale, de rendez-vous en attitude, de détection de la couverture nuageuse, d'observation de zones au sol, de télédéchargement de données, de rechargement des batteries et de pointage géocentrique. De nombreuses simulations hors ligne et en ligne nous ont finalement permis de montrer l'intérêt de détecter la couverture nuageuse en avant du satellite et de mettre en place des mécanismes de décision et d'optimisation à bord des futurs satellites d'observation.

Contributions

D'un point de vue général, cette thèse a permis de montrer qu'il est désormais envisageable de mettre en place des mécanismes de contrôle autonomes pour la gestion de missions spatiales aussi complexes qu'une mission d'observation de la Terre par un satellite agile équipé d'un instrument de détection de la couverture nuageuse.

Problème de décision

Le problème de décision auquel nous nous sommes intéressés est un problème d'optimisation du retour global d'une mission d'observation de la Terre. Ce problème a été formalisé comme un problème de décision séquentielle : les décisions sont prises en séquence, après chaque fin d'action. Afin de maximiser le retour d'une telle mission nous avons choisi de suivre une approche purement réactive consistant à déterminer à chaque instant la meilleure prochaine action que le satellite doit engager.

La taille de l'espace de recherche et les incertitudes liées à la mission nous ont orientés vers :

- un mécanisme de prise de décision prenant en compte toutes les informations disponibles sur l'état courant du satellite, les observations envisageables à partir de l'état courant, les prévisions météorologiques et les informations issues de la détection de la couverture nuageuse ;
- un raisonnement sur un horizon limité ;
- un oubli volontaire des incertitudes dans le raisonnement et une remise en cause de la décision à chaque fois que la fin de l'horizon est atteinte ou que l'état courant est modifié.

Un soin particulier a été apporté à la modélisation du problème dans le cadre CNT (Constraint Network on Timelines). Cette modélisation a servi de base aux différents algorithmes de décision mis en place par la suite.

Calcul de trajectoires en attitude

Au cours de son raisonnement, afin d'estimer notamment les productions et consommations d'énergie, la tâche délibérative du logiciel de vol du satellite a besoin de calculer les trajectoires en attitude à suivre au cours de chaque action envisageable sur l'horizon de planification. À cause des puissances de calcul limitées des engins spatiaux actuels, il n'est pas envisageable d'embarquer les méthodes de calculs exactes de ces trajectoires, actuellement utilisées au sol. Nous avons donc proposé une méthode d'estimation de ces trajectoires à bord, reposant sur l'utilisation d'un solveur de CSP continu sur un modèle simplifié de la cinématique du satellite considéré. Même si cette méthode ne permet toujours pas de calculer des trajectoires en attitude optimales en temps raisonnable, l'utilisation d'heuristiques pour guider les calculs et l'optimisation possible de ces calculs laissent à penser que ce pourrait bientôt être le cas.

Architecture de contrôle autonome

Pour le contrôle du satellite, nous avons adapté au contexte de la mission d'observation, étendu et amélioré l'architecture générique réactive - délibérative développée dans le cadre du projet AGATA. L'adaptation a notamment consisté à prendre en compte de nouveaux types de données apparaissant dans le contexte de la mission d'observation. Ces types de

données sont par exemple des informations sur la couverture nuageuse reçues après une action de détection, des requêtes d'observation envoyées par les centres de mission, ou des manœuvres orbitales générées par le Contrôle d'Orbite Autonome du satellite.

L'extension de cette architecture générique a quant à elle, consisté à enrichir la description d'un état du système pour qu'elle puisse représenter l'état complet du satellite dans son environnement, et à doter le satellite d'un modèle complet du système (prenant en compte le calcul des trajectoires en attitude) sur lequel les tâches de raisonnement s'appuieront.

Enfin plusieurs améliorations ont été ajoutées à l'architecture de contrôle, notamment au niveau du comportement de la tâche délibérative qui, à chaque fois qu'elle est relancée suite à une prise de décision, redémarre une nouvelle recherche de la prochaine action optimale à entreprendre, non plus en repartant de zéro, mais en tenant compte du raisonnement réalisé au cours de la recherche de l'action optimale précédente.

Une fois cette architecture mise en place, nous avons proposé plusieurs règles de décision, dont deux ont été implémentées (DR1 et DR2), pour la partie réactive du logiciel de vol du satellite. Du côté de la partie délibérative, un algorithme glouton stochastique itéré a été défini et implémenté.

Environnement de simulation

Afin de tester le comportement de cette architecture de contrôle autonome dans le contexte réaliste d'une mission d'observation de la Terre, nous avons développé un environnement de simulation complet d'une mission d'observation de la Terre par un satellite agile. Cet environnement nous a permis de simuler en temps-réel le comportement d'un satellite doté d'une telle architecture de contrôle et de comparer le retour de la mission gérée de façon autonome à bord du satellite et le retour de la mission telle qu'elle est gérée aujourd'hui pour les satellites actuels d'observation (plans générés au sol et téléchargés quotidiennement au satellite).

Cet environnement qui permet de visualiser en parallèle l'évolution temps-réel du système et le comportement du logiciel de vol du satellite (prises de décisions réactives, calculs des échéances, construction des délibérations) pourra par la suite être mis à la disposition de la communauté scientifique afin qu'elle puisse tester différents mécanismes réactifs et algorithmes d'optimisation.

Perspectives

Si les possibilités d'amélioration des mécanismes de prise de décision réactifs développés semblent restreintes, il reste encore de nombreuses voies à explorer pour ce qui est des algorithmes d'optimisation utilisés dans la tâche délibérative. Il serait en effet intéressant d'utiliser des algorithmes de programmation dynamique tel que cela a été fait dans [Damiani, 2005], des méthodes de recherche locale, ou bien d'introduire des mécanismes d'ap-

prentissage au niveau de la résolution d'une instance de problème par un algorithme glouton stochastique itéré. On pense notamment à des algorithmes de type "colonies de fourmis" qui permettent de privilégier des chemins dans l'arbre de recherche en affectant un poids à chacun de ses nœuds, et en les modifiant en fonction des valeurs des différents chemins parcourus.

Pour la gestion de la mission d'observation nous avons choisi une approche purement réactive : c'est la tâche réactive de l'architecture mise en place qui joue le rôle d'interface entre le satellite et son environnement, lui permettant de répondre instantanément aux évènements. Cette composante réactive contrôle une tâche délibérative qui, lorsqu'elle a le temps nécessaire, tente d'optimiser le choix des décisions prises. En supposant que la plupart des évènements extérieurs auxquels le satellite doit réagir sont les arrivées de nouvelles informations sur la couverture nuageuse et les arrivées de nouvelles requêtes d'observation, il serait intéressant de tester des approches plus proactives en prévoyant, dans le raisonnement, les arrivées de nouvelles requêtes (qui ont lieu lorsque le satellite passe en visibilité d'un centre de mission) ou les informations sur la couverture nuageuse (il y a par exemple des zones sur Terre ou la couverture nuageuse est moins sujette à variations que d'autres).

Tout au long de notre étude, nous avons pu nous intéresser à deux types d'approches de résolution du problème de décision : une approche à base de règles de décision écrites "à la main", et une approche à base de modèle utilisé pour une prise de décision en ligne. Même si ces deux approches qui nous ont paru les plus appropriées pour le type de problème auquel nous étions confrontés, il resterait encore à explorer une approche à base de modèle utilisé pour une prise de décision hors ligne ou une approche à base de règles de décision couplées à des mécanismes d'apprentissage permettant d'améliorer ces règles de décision en fonction des situations rencontrées.

Un aspect de la mission dont nous n'avons pas beaucoup tiré parti dans cette étude, est le caractère cyclique d'une mission satellitaire d'observation de la Terre : tout au long de sa vie, le satellite alterne entre une période d'éclairement pendant laquelle il réalise des observations, et une période d'éclipse au cours de laquelle il s'occupe principalement des télédéchargements de données. Il serait intéressant de s'attacher à la définition de stratégies de haut niveau pour la gestion de la mission. Étant donné qu'en période d'éclipse le satellite n'est soumis qu'à très peu d'évènements extérieurs (pas d'observation ou de détection de la couverture nuageuse), on pourrait imaginer que le satellite mette à profit ces temps de latence en construisant des plans (éventuellement conditionnels pour prendre en compte les résultats des actions de détection) sur la prochaine demi-orbite éclairée, qui serviraient de base aux raisonnements en période d'éclairement.

Cette thèse s'est articulée autour d'une mission de référence d'observation de la Terre. Mais elle s'inscrit aussi dans un projet plus vaste portant sur l'autonomie des engins spatiaux. Nous avons validé l'utilisation d'une architecture de contrôle autonome dans le cas particulier d'une mission d'observation de la Terre par un satellite d'observation agile, et montré son intérêt dans l'amélioration du retour de la mission. Il serait maintenant

intéressant d'appliquer les mécanismes que nous avons développés à d'autres scénarios comme celui d'un rover martien, ou d'une sonde d'exploration spatiale pour lesquels la dynamique du système et les contraintes environnementales sont différentes. Il serait alors utile d'étendre les mécanismes de décision développés, à la gestion de plusieurs séquences d'activités concurrentes, et non plus les restreindre à une seule timeline comme cela a été le cas dans ces travaux de thèse.

Annexe A

Code source RealPaver

1 Minimisation de la durée d'un rendez-vous

```
/* Minimisation de la duree d'un rendez-vous en attitude */
/* Attitude de référence 1 */

/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
Bisection

        choice = tf ,                                   /* parametres de la recherche */
        parts = 2 ,
        precision = 10000 ,
        number = 1 ;

/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
Constants

        THETAxd = 1.3411881465837094 ,                  /* angle de roulis initial du satellite */
        OMEGAxd = 0.04953477212920501 ,                 /* vitesse en roulis initiale du satellite */
        THETAyd = -0.06824842792507001 ,                /* angle de tangage initial du satellite */
        OMEGAyd = 0.013343892193639864 ,                /* vitesse en tangage initiale du satellite */
        THETAzd = -0.2538169282914785 ,                 /* angle de lacet initial du satellite */
        OMEGAzd = 0.04939014694826076 ,                 /* vitesse en lacet initiale du satellite */
        latM = 0.6981317007977318 ,                     /* latitude du point vise, M */
        asMd = -1.3009576317568108 ,                    /* ascension droite du point M a Td */
        cap = 0.5235987755982988 ,                      /* cap de l'observation a rallier */
        vbal = 6586.562758305667 ,                      /* vitesse de balayage au cours de l'observation */
        Ix = 850 ,                                      /* inertie du satellite en roulis */
        Iy = 850 ,                                      /* inertie du satellite en tangage */
        Iz = 750 ,                                      /* inertie du satellite en lacet */
        omegaTerre = 2*@pi/86400 ,                      /* vitesse de rotation de la Terre sur elle-même */
        RT = 6378136.6 ,                                /* rayon terrestre */
        RO = 7204793.688912138 ,                        /* rayon de l'orbite */
        Omegad = 0.9106953503981212 ,                   /* asc. droite du noeud ascendant à Td */
        OmegaPoint = 1.99322968461429e-7 ,              /* vitesse de rotation du noeud ascendant */
        nuPoint = 0.0010326782211446627 ,               /* mouvement moyen du satellite sur son orbite */
        nud = 2.923683061 ,                             /* anomalie vraie du satellite à Td */
        i = 1.7230448881048661 ,                        /* inclinaison de l'orbite du satellite */
        Td = 28164.0 ,                                  /* date de debut du rendez-vous */
        Ts = 28164.0 ,                                  /* date de debut de la fenetre de visibilité de la zone */
        Te = 28264.0 ,                                  /* date de fin de la fenetre de visibilité de la zone */
        HmodMAX = 60 ,                                  /* moment maximal en module */
        HxMAX = 45 ,                                    /* moment maximal dans le repere satellite */
        CxMAX = 7 ,                                     /* couple maximal dans le repere satellite */
        HyMAX = 45 ,                                    /* moment maximal dans le repere satellite */
        CyMAX = 7 ,                                     /* couple maximal dans le repere satellite */
        HzMAX = 20 ,                                    /* moment maximal dans le repere satellite */
        CzMAX = 6 ;                                     /* couple maximal dans le repere satellite */
```

```
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
Variables

        real t in [Ts, Te] : 0.1,                       /* date de fin du changement d'attitude */
        real asM in [asMd, asMd + omegaTerre*Te] ,      /* ascension droite du point M */
        real xM in [-RT, RT] ,                          /* coordonnees cartesiennes du point M */
        real yM in [-RT, RT] ,
        real zM in [-RT, RT] ,
        real xS in [-RO, RO] ,                          /* coordonnees cartesiennes du satellite */
        real yS in [-RO, RO] ,
        real zS in [-RO, RO] ,
        real xa1 in [-1, 1] ,                           /* definition du repere lie a la zone */
        real xa2 in [-1, 1] ,
        real xa3 in [-1, 1] ,
        real ya1 in [-1, 1] ,
        real ya2 in [-1, 1] ,
        real ya3 in [-1, 1] ,
        real za1 in [-1, 1] ,
        real za2 in [-1, 1] ,
        real za3 in [-1, 1] ,
        real zol1 in [-1, 1] ,                          /* definition du repere orbital local */
        real zol2 in [-1, 1] ,
        real zol3 in [-1, 1] ,
        real vMx in ]-oo, +oo[ ,                        /* vitesse du point vise, M */
        real vMy in ]-oo, +oo[ ,
        real vMz in ]-oo, +oo[ ,
        real rotSx in [-nuPoint, nuPoint] ,             /* vitesse de rotation du satellite par rapport a Rs, dans Rs */
        real rotSy in [-nuPoint, nuPoint] ,
        real rotSz in [OmegaPoint-nuPoint, OmegaPoint+nuPoint] ,
        real vSx in ]-oo, +oo[ ,                        /* vitesse de deplacement du satellite sur son orbite */
        real vSy in ]-oo, +oo[ ,
        real vSz in ]-oo, +oo[ ,
        real zs1 in [-1, 1] ,                           /* definition du repere satellite */
        real zs2 in [-1, 1] ,
        real zs3 in [-1, 1] ,
        real ys1 in [-1, 1] ,
        real ys2 in [-1, 1] ,
        real ys3 in [-1, 1] ,
        real xs1 in [-1, 1] ,
        real xs2 in [-1, 1] ,
        real xs3 in [-1, 1] ,
        real omegaxf in ]-oo, +oo[ ,                    /* vitesse de rotation du satellite par rapport à Rc, dans Rc */
        real omegayf in ]-oo, +oo[ ,
        real omegazf in ]-oo, +oo[ ,
        real omegaxs in [-HxMAX/Ix, HxMAX/Ix] ,         /* vitesse de rotation du satellite par rapport à Rc, dans Rs */
        real omegays in [-HyMAX/Iy, HyMAX/Iy] ,
        real omegazs in [-HzMAX/Iz, HzMAX/Iz] ,
        real Omega in [Omegad, Omegad + OmegaPoint*(Te-Td)] ,   /* position du noeud ascendant de l'orbite */
        real nu in [nud, nud + nuPoint*(Te-Td)] ,       /* anomalie vraie du satellite */
        real thetaxf in ]-@pi, @pi] ,                   /* angle de roulis de fin */
        real thetayf in ]-@pi, @pi] ,                   /* angle de tangage de fin */
        real thetazf in ]-@pi, @pi] ,                   /* angle de lacet de fin */
        real cx1 in [-CxMAX, +CxMAX] ,                  /* couple de roulis au cours de la première phase */
        real cx2 in [-CxMAX, +CxMAX] ,                  /* couple de roulis au cours de la dernière phase */
        real cy1 in [-CyMAX, +CyMAX] ,                  /* couple de tangage au cours de la première phase */
        real cy2 in [-CyMAX, +CyMAX] ,                  /* couple de tangage au cours de la dernière phase */
        real cz1 in [-CzMAX, +CzMAX] ,                  /* couple de lacet au cours de la première phase */
        real cz2 in [-CzMAX, +CzMAX] ,                  /* couple de lacet au cours de la dernière phase */
        real hxp in [-HxMAX, +HxMAX] ,                  /* moment en roulis au cours du palier */
        real hyp in [-HyMAX, +HyMAX] ,                  /* moment en tangage au cours du palier */
        real hzp in [-HzMAX, +HzMAX] ,                  /* moment en lacet au cours du palier */
        real t1 in [Td, Te] ,                           /* date de fin de la première phase */
        real t2 in [Td, Te] ;                           /* date de début de la dernière phase */

/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
/*****************************************************************************************************************************/
Constraints

        /* Contraintes physiques */
        t2 <= t ,
        t1 <= t2 ,

        /* Evolution du repere cible Ra */
        xa1 =-cos(asM)*sin(latM)*cos(cap) + sin(asM)*sin(cap) ,
        xa2 =-sin(asM)*sin(latM)*cos(cap) - cos(asM)*sin(cap) ,
        xa3 =cos(latM)*cos(cap) ,
        ya1 =cos(asM)*sin(latM)*sin(cap) + sin(asM)*cos(cap) ,
        ya2 =sin(asM)*sin(latM)*sin(cap) - cos(asM)*cos(cap) ,
        ya3 =-cos(latM)*sin(cap) ,
        za1 =cos(asM)*cos(latM) ,
```

```
za2 =sin(asM)*cos(latM) ,
za3 =sin(latM) ,

/* Evolution de la position du point M */
asM = asMd + omegaTerre*(t-Td) ,
xM = RT*za1 ,
yM = RT*za2 ,
zM = RT*za3 ,

/* Vitesse de M dans Rc */
vMx = -yM*omegaTerre ,
vMy = xM*omegaTerre ,
vMz = 0 ,

/* Evolution du repere orbital local */
zol1 =-sin(Omega)*cos(i)*sin(nu) + cos(Omega)*cos(nu) ,
zol2 =cos(Omega)*cos(i)*sin(nu) + sin(Omega)*cos(nu) ,
zol3 =sin(i)*sin(nu) ,

/* Evolution de la position du satellite S */
Omega = Omegad + OmegaPoint*(t-Td) ,
nu = nud + nuPoint*(t-Td) ,
xS = RO*zol1 ,
yS = RO*zol2 ,
zS = RO*zol3 ,

/* Vecteur rotation du satellite sur son orbite dans Rc */
rotSx = nuPoint*sin(i)*sin(Omega) ,
rotSy = -nuPoint*sin(i)*sin(Omega) ,
rotSz = OmegaPoint + nuPoint*cos(i) ,

/* Vitesse du satellite dans Rc */
vSx = zS*rotSy - yS*rotSz ,
vSy = xS*rotSz - zS*rotSx ,
vSz = yS*rotSx - xS*rotSy ,

/* Evolution du repere satellite en pointage de M */
zs1 =(xM-xS)/sqrt((xM-xS)^2+(yM-yS)^2+(zM-zS)^2) ,
zs2 =(yM-yS)/sqrt((xM-xS)^2+(yM-yS)^2+(zM-zS)^2) ,
zs3 =(zM-zS)/sqrt((xM-xS)^2+(yM-yS)^2+(zM-zS)^2) ,
ys1 = zs2*xs3 - zs3*xs2 ,
ys2 = zs3*xs1 - zs1*xs3 ,
ys3 = zs1*xs2 - zs2*xs1 ,
xs1 = ya2*zs3 - ya3*zs2 ,
xs2 = ya3*zs1 - ya1*zs3 ,
xs3 = ya1*zs2 - ya2*zs1 ,

/* Evolution de l'angle de roulis de fin en pointage de M */
thetaxf = -atan(zs2/zs3) ,

/* Evolution de la vitesse de roulis de fin en pointage de M (où omega est le vecteur vitesse de rotation) */
omegaxf = (vSy + omegazf*(xM-xS) - vMy - vbal*ya1)/(zM-zS) ,

/* Evolution de l'angle de tangage de fin */
thetayf = asin(zs1) ,

/* Evolution de la vitesse de tangage de fin en pointage de M (où omega est le vecteur vitesse de rotation) */
omegayf = ((vMx + vbal*xa1 - vSx) + (yM-yS)*(vMy + vbal*ya1 - vSy + (zM-zS)*(vMz + vbal*za1 - vSz)))
          / ((zM-zS) - (yM-yS)*(zM-zS)*(xM-xS)) ,

/* Evolution de l'angle de lacet de fin */
thetazf = -atan(ys1/xs1) ,

/* Evolution de la vitesse de lacet de fin en pointage de M (où omega est le vecteur vitesse de rotation) */
omegazf = (vSx + omegayf*(zM-zS) - vMx - vbal*xa1)/(yM-yS) ,

/* Calcul du vecteur vitesse de rotation du satellite dans le repère satellite */
omegaxf = xs1*omegaxs + ys1*omegays + zs1*omegazs ,
omegayf = xs2*omegaxs + ys2*omegays + zs2*omegazs ,
omegazf = xs3*omegaxs + ys3*omegays + zs3*omegazs ,

/* Continuité du moment cinétique en roulis en t1 et t2 */
hxp = Ix*OMEGAxd + cx1*(t1-Td) ,
hxp = Ix*omegaxs + cx2*(t2-t) ,

/* Orientation du satellite en roulis à la date de fin */
Ix*thetaxf = Ix*THETAxd + (Ix*OMEGAxd-cx1*Td)*(t1-Td) + 1/2*cx1*(t1^2-Td^2) + (Ix*omegaxs-cx2*t)*(t-t2)
             + 1/2*cx2*(t^2-t2^2) + hxp*(t2-t1) ,

/* Continuité du moment cinétique en tangage en t1 et t2 */
hyp = Iy*OMEGAyd + cy1*(t1-Td) ,
hyp = Iy*omegays + cy2*(t2-t) ,

/* Orientation du satellite en tangage à la date de fin */
Iy*thetayf = Iy*THETAyd + (Iy*OMEGAyd-cy1*Td)*(t1-Td) + 1/2*cy1*(t1^2-Td^2) + (Iy*omegays-cy2*t)*(t-t2)
```

```
                + 1/2*cy2*(t^2-t2^2) + hyp*(t2-t1) ,

        /* Continuité du moment cinétique en lacet en t1 et t2 */
        hzp = Iz*OMEGAzd + cz1*(t1-Td) ,
        hzp = Iz*omegazs + cz2*(t2-t) ,

        /* Orientation du satellite en lacet à la date de fin */
        Iz*thetazf = Iz*THETAzd + (Iz*OMEGAzd-cz1*Td)*(t1-Td) + 1/2*cz1*(t1^2-Td^2) + (Iz*omegazs-cz2*t)*(t-t2)
                + 1/2*cz2*(t^2-t2^2) + hzp*(t2-t1) ;
```

2 Minimisation de l'angle de prise de vue

```
/* Minimisation de l'angle de prise de vue */
/* Attitude de référence 1 */

/*********************************************************************************************************************/
/*********************************************************************************************************************/
/*********************************************************************************************************************/
/*********************************************************************************************************************/
Output

        digits = 4 ,                                    /* parametres de la recherche */
        mode = hull ;

Bisection

        choice = tf ,
        parts = 2 ,
        precision = 10000 ,
        number = 1 ;

Consistency

        local = hc4 ;
/*********************************************************************************************************************/
/*********************************************************************************************************************/
/*********************************************************************************************************************/
/*********************************************************************************************************************/
Constants

        THETAxd = 1.3411881465837094 ,                  /* angle de roulis initial du satellite */
        OMEGAxd = 0.04953477212920501 ,                 /* vitesse en roulis initiale du satellite */
        THETAyd = -0.06824842792507001 ,                /* angle de tangage initial du satellite */
        OMEGAyd = 0.013343892193639864 ,                /* vitesse en tangage initiale du satellite */
        THETAzd = -0.2538169282914785 ,                 /* angle de lacet initial du satellite */
        OMEGAzd = 0.04939014694826076 ,                 /* vitesse en lacet initiale du satellite */
        latM = 0.6981317007977318 ,                     /* latitude du point vise, M */
        asMd = -1.3009576317568108 ,                    /* ascension droite du point M a Td */
        cap = 0.5235987755982988 ,                      /* cap de l'observation a rallier */
        vbal = 6586.562758305667 ,                      /* vitesse de balayage au cours de l'observation */
        Ix = 850 ,                                      /* inertie du satellite en roulis */
        Iy = 850 ,                                      /* inertie du satellite en tangage */
        Iz = 750 ,                                      /* inertie du satellite en lacet */
        omegaTerre = 2*@pi/86400 ,                      /* vitesse de rotation de la Terre sur elle-même */
        RT = 6378136.6 ,                                /* rayon terrestre */
        RO = 7204793.688912138 ,                        /* rayon de l'orbite */
        Omegad = 0.9106953503981212 ,                   /* asc. droite du noeud ascendant à Td */
        OmegaPoint = 1.99322968461429e-7 ,              /* vitesse de rotation du noeud ascendant */
        nuPoint = 0.0010326782211446627 ,               /* mouvement moyen du satellite sur son orbite */
        nud = 2.923683061 ,                             /* anomalie vraie du satellite à Td */
        i = 1.7230448881048661 ,                        /* inclinaison de l'orbite du satellite */
        Td = 28164.0 ,                                  /* date de debut du rendez-vous */
        Ts = 28164.0 ,                                  /* date de debut de la fenetre de visibilité de la zone */
        Te = 28264.0 ,                                  /* date de fin de la fenetre de visibilité de la zone */
        HmodMAX = 60 ,                                  /* moment maximal en module */
        HxMAX = 45 ,                                    /* moment maximal dans le repere satellite */
        CxMAX = 7 ,                                     /* couple maximal dans le repere satellite */
        HyMAX = 45 ,                                    /* moment maximal dans le repere satellite */
        CyMAX = 7 ,                                     /* couple maximal dans le repere satellite */
        HzMAX = 20 ,                                    /* moment maximal dans le repere satellite */
        CzMAX = 6 ;                                     /* couple maximal dans le repere satellite */

/*********************************************************************************************************************/
/*********************************************************************************************************************/
/*********************************************************************************************************************/
/*********************************************************************************************************************/
Variables

        real t in [1, +oo[ : 0.01,                      /* variable à minimiser
```

```
                                                    (ici l'angle entre la direction de zs et celle de zol) */
        real temps in [Ts, Te] : 0.01,              /* date de fin du changement d'attitude */
        real asM in [asMd, asMd + omegaT*Te] ,      /* ascension droite du point M */
        real xM in [-RT, RT] ,                      /* coordonnees cartesiennes du point M */
        real yM in [-RT, RT] ,
        real zM in [-RT, RT] ,
        real xS in [-RO, RO] ,                      /* coordonnees cartesiennes du satellite */
        real yS in [-RO, RO] ,
        real zS in [-RO, RO] ,
        real xa1 in [-1, 1] ,                       /* definition du repere lie a la zone */
        real xa2 in [-1, 1] ,
        real xa3 in [-1, 1] ,
        real ya1 in [-1, 1] ,
        real ya2 in [-1, 1] ,
        real ya3 in [-1, 1] ,
        real za1 in [-1, 1] ,
        real za2 in [-1, 1] ,
        real za3 in [-1, 1] ,
        real zol1 in [-1, 1] ,                      /* definition du repere orbital local */
        real zol2 in [-1, 1] ,
        real zol3 in [-1, 1] ,
        real vMx in ]-oo, +oo[ ,                    /* vitesse du point vise, M */
        real vMy in ]-oo, +oo[ ,
        real vMz in ]-oo, +oo[ ,
        real rotSx in [-nuPoint, nuPoint] ,         /* vitesse de rotation du satellite par rapport a Rs, dans Rs */
        real rotSy in [-nuPoint, nuPoint] ,
        real rotSz in [OmegaPoint-nuPoint, OmegaPoint+nuPoint] ,
        real vSx in ]-oo, +oo[ ,                    /* vitesse de deplacement du satellite sur son orbite */
        real vSy in ]-oo, +oo[ ,
        real vSz in ]-oo, +oo[ ,
        real zs1 in [-1, 1] ,                       /* definition du repere satellite */
        real zs2 in [-1, 1] ,
        real zs3 in [-1, 1] ,
        real ys1 in [-1, 1] ,
        real ys2 in [-1, 1] ,
        real ys3 in [-1, 1] ,
        real xs1 in [-1, 1] ,
        real xs2 in [-1, 1] ,
        real xs3 in [-1, 1] ,
        real omegaxf in ]-oo, +oo[ ,                /* vitesse de rotation du satellite par rapport à Rc, dans Rc */
        real omegayf in ]-oo, +oo[ ,
        real omegazf in ]-oo, +oo[ ,
        real omegaxs in [-HxMAX/Ix, HxMAX/Ix] ,     /* vitesse de rotation du satellite par rapport à Rc, dans Rs */
        real omegays in [-HyMAX/Iy, HyMAX/Iy] ,
        real omegazs in [-HzMAX/Iz, HzMAX/Iz] ,
        real Omega in [Omegad, Omegad + OmegaPoint*(Te-Td)] ,   /* position du noeud ascendant de l'orbite */
        real nu in [nud, nud + nuPoint*(Te-Td)] ,   /* anomalie vraie du satellite */
        real delta_thetazf in ]-@pi, @pi] ,         /* débattement angulaire en lacet à la fin */
        real delta_thetayf in ]-@pi, @pi] ,         /* débattement angulaire en tangage à la fin */
        real delta_thetaxf in ]-@pi, @pi] ,         /* débattement angulaire en roulis à la fin */
        real M11 in ]-oo, +oo[ ,                    /* coefficient de la matrice de passage M */
        real M21 in ]-oo, +oo[ ,                    /* coefficient de la matrice de passage M */
        real M31 in ]-oo, +oo[ ,                    /* coefficient de la matrice de passage M */
        real M32 in ]-oo, +oo[ ,                    /* coefficient de la matrice de passage M */
        real M33 in ]-oo, +oo[ ,                    /* coefficient de la matrice de passage M */
        real cx1 in [-CxMAX, +CxMAX] ,              /* couple de roulis au cours de la première phase */
        real cx2 in [-CxMAX, +CxMAX] ,              /* couple de roulis au cours de la dernière phase */
        real cy1 in [-CyMAX, +CyMAX] ,              /* couple de tangage au cours de la première phase */
        real cy2 in [-CyMAX, +CyMAX] ,              /* couple de tangage au cours de la dernière phase */
        real cz1 in [-CzMAX, +CzMAX] ,              /* couple de lacet au cours de la première phase */
        real cz2 in [-CzMAX, +CzMAX] ,              /* couple de lacet au cours de la dernière phase */
        real hxp in [-HxMAX, +HxMAX] ,              /* moment en roulis au cours du palier */
        real hyp in [-HyMAX, +HyMAX] ,              /* moment en tangage au cours du palier */
        real hzp in [-HzMAX, +HzMAX] ,              /* moment en lacet au cours du palier */
        real t1 in [Td, Te] ,                       /* date de fin de la première phase */
        real t2 in [Td, Te] ;                       /* date de début de la dernière phase */

/*****************************************************************************************************************/
/*****************************************************************************************************************/
/*****************************************************************************************************************/
/*****************************************************************************************************************/
Constraints

        cx1 = -CxMAX ,
        cx2 = CxMAX ,

        /* variable à minimiser : l'inverse du cosinus de l'angle entre zs et zol */
        t = 1/(zs1*zol1 + zs2*zol2 + zs3*zol3) ,

        /* Contraintes physiques */
        t2 <= temps ,
        t1 <= t2 ,

        /* Evolution du repere cible Ra */
        xa1 =-cos(asM)*sin(latM)*cos(cap) + sin(asM)*sin(cap) ,
```

```
xa2 =-sin(asM)*sin(latM)*cos(cap) - cos(asM)*sin(cap) ,
xa3 =cos(latM)*cos(cap) ,
ya1 =cos(asM)*sin(latM)*sin(cap) + sin(asM)*cos(cap) ,
ya2 =sin(asM)*sin(latM)*sin(cap) - cos(asM)*cos(cap) ,
ya3 =-cos(latM)*sin(cap) ,
za1 =cos(asM)*cos(latM) ,
za2 =sin(asM)*cos(latM) ,
za3 =sin(latM) ,

/* Evolution de la position du point M */
asM = asMd + omegaT*(temps-Td) ,
xM = RT*za1 ,
yM = RT*za2 ,
zM = RT*za3 ,

/* Vitesse de M dans Rc */
vMx = -yM*omegaT ,
vMy = xM*omegaT ,
vMz = 0 ,

/* Evolution du repere orbital local */
zol1 =-sin(Omega)*cos(i)*sin(nu) + cos(Omega)*cos(nu) ,
zol2 =cos(Omega)*cos(i)*sin(nu) + sin(Omega)*cos(nu) ,
zol3 =sin(i)*sin(nu) ,

/* Evolution de la position du satellite S */
Omega = Omegad + OmegaPoint*(temps-Td) ,
nu = nud + nuPoint*(temps-Td) ,
xS = RO*zol1 ,
yS = RO*zol2 ,
zS = RO*zol3 ,

/* Vecteur rotation du satellite sur son orbite dans Rc */
rotSx = nuPoint*sin(i)*sin(Omega) ,
rotSy = -nuPoint*sin(i)*cos(Omega) ,
rotSz = OmegaPoint + nuPoint*cos(i) ,

/* Vitesse du satellite dans Rc */
vSx = zS*rotSy - yS*rotSz ,
vSy = xS*rotSz - zS*rotSx ,
vSz = yS*rotSx - xS*rotSy ,

/* Evolution du repere satellite en pointage de M */
zs1 = (xS-xM)/sqrt((xM-xS)^2+(yM-yS)^2+(zM-zS)^2) ,
zs2 = (yS-yM)/sqrt((xM-xS)^2+(yM-yS)^2+(zM-zS)^2) ,
zs3 = (zS-zM)/sqrt((xM-xS)^2+(yM-yS)^2+(zM-zS)^2) ,
ys1 = zs2*xs3 - zs3*xs2 ,
ys2 = zs3*xs1 - zs1*xs3 ,
ys3 = zs1*xs2 - zs2*xs1 ,
xs1 = (ya2*zs3 - ya3*zs2)/sqrt((ya2*zs3 - ya3*zs2)^2 + (ya3*zs1 - ya1*zs3)^2 + (ya1*zs2 - ya2*zs1)^2) ,
xs2 = (ya3*zs1 - ya1*zs3)/sqrt((ya2*zs3 - ya3*zs2)^2 + (ya3*zs1 - ya1*zs3)^2 + (ya1*zs2 - ya2*zs1)^2) ,
xs3 = (ya1*zs2 - ya2*zs1)/sqrt((ya2*zs3 - ya3*zs2)^2 + (ya3*zs1 - ya1*zs3)^2 + (ya1*zs2 - ya2*zs1)^2) ,

/* Coefficiants utiles de la matrice de passage M */
M11 = cos(THETAzd)*cos(THETAyd)*xs1 + sin(THETAzd)*cos(THETAyd)*ys1 - sin(THETAyd)*zs1 ,
M21 =cos(THETAzd)*cos(THETAyd)*xs2 + sin(THETAzd)*cos(THETAyd)*ys2 - sin(THETAyd)*zs2 ,
M31 =cos(THETAzd)*cos(THETAyd)*xs3 + sin(THETAzd)*cos(THETAyd)*ys3 - sin(THETAyd)*zs3 ,
M32 = (-sin(THETAzd)*cos(THETAxd)+cos(THETAzd)*sin(THETAyd)*sin(THETAxd))*xs3
      + (cos(THETAzd)*cos(THETAxd)+sin(THETAzd)*sin(THETAyd)*sin(THETAxd))*ys3
      + cos(THETAyd)*sin(THETAxd)*zs3 ,
M33 = (sin(THETAzd)*sin(THETAxd)+cos(THETAzd)*sin(THETAyd)*cos(THETAxd))*xs3
      + (-cos(THETAzd)*sin(THETAxd)+sin(THETAzd)*sin(THETAyd)*cos(THETAxd))*ys3
      + cos(THETAyd)*cos(THETAxd)*zs3 ,

/* Evolution du débattement angulaire en lacet en pointage de M */
delta_thetazf = atan(M21/M11) ,

/* Evolution du débattement angulaire en tangage en pointage de M */
delta_thetayf = -asin(M31) ,

/* Evolution du débattement angulaire en roulis en pointage de M */
delta_thetaxf = atan(M32/M33) ,

/* La composante selon xa de la vitesse du point visé est égale à vbal + la composante selon xa de VM */
omegaxf*((yM-yS)*xa3-(zM-zS)*xa2) + omegayf*((zM-zS)*xa1-(xM-xS)*xa3) + omegazf*((xM-xS)*xa2-(yM-yS)*xa1)
= vbal + (vMx-vSx)*xa1 + (vMy-vSy)*xa2 + (vMz-vSz)*xa3 ,

/* La composante selon ya de la vitesse du point visé est égale à la composante selon ya de VM */
omegaxf*((yM-yS)*ya3-(zM-zS)*ya2) + omegayf*((zM-zS)*ya1-(xM-xS)*ya3) + omegazf*((xM-xS)*ya2-(yM-yS)*ya1)
= (vMx-vSx)*ya1 + (vMy-vSy)*ya2 + (vMz-vSz)*ya3 ,

/* On impose l'angle (donc le produit scalaire) entre omegaf et SM : pour supprimer le degré de liberté
   supplémentaire */
omegaxf*(xM-xS) + omegayf*(yM-yS) + omegazf*(zM-zS) = 1 ,
```

```
/* Calcul du vecteur vitesse de rotation du satellite dans le repère satellite */
omegaxf = xs1*omegaxs + ys1*omegays + zs1*omegazs ,
omegayf = xs2*omegaxs + ys2*omegays + zs2*omegazs ,
omegazf = xs3*omegaxs + ys3*omegays + zs3*omegazs ,

/* Continuité du moment cinétique en roulis en t1 et t2 */
hxp = Ix*OMEGAxd + cx1*(t1-Td) ,
hxp = Ix*omegaxs + cx2*(t2-temps) ,

/* Orientation du satellite en roulis à la date de fin */
Ix*delta_thetaxf = Ix*OMEGAxd*(t1-Td)+1/2*cx1*(t1-Td)^2 + Ix*omegaxs*(temps-t2)+1/2*cx2*(temps-t2)^2
                   + hxp*(t2-t1) ,

/* Continuité du moment cinétique en tangage en t1 et t2 */
hyp = Iy*OMEGAyd + cy1*(t1-Td) ,
hyp = Iy*omegays + cy2*(t2-temps) ,

/* Orientation du satellite en tangage à la date de fin */
Iy*delta_thetayf = Iy*OMEGAyd*(t1-Td)+1/2*cy1*(t1-Td)^2 + Iy*omegays*(temps-t2)+1/2*cy2*(temps-t2)^2
                   + hyp*(t2-t1) ,

/* Continuité du moment cinétique en lacet en t1 et t2 */
hzp = Iz*OMEGAzd + cz1*(t1-Td) ,
hzp = Iz*omegazs + cz2*(t2-temps) ,

/* Orientation du satellite en lacet à la date de fin */
Iz*delta_thetazf = Iz*OMEGAzd*(t1-Td)+1/2*cz1*(t1-Td)^2 + Iz*omegazs*(temps-t2)+1/2*cz2*(temps-t2)^2
                   + hzp*(t2-t1) ;
```

Annexe B

Code source de la tâche réactive

```
module DECISIONNEL_REACTIF :

%%%=== types, constantes, fonctions, procedures et taches externes

type Action, BasicObservation, CloudCoverInformation, Decision, Manoeuvre, Plan, RequestPack, State;

constant DECISION_VIDE : Decision,
         TRES_LOINTAIN : double;

function        initialRealState() : State,
                initialInnerState() : State,
                initialDecisionState() : State,
                initialTime() : double,
                initialPlan() : Plan,
                getTime(double) : double,
                etat_reel(double) : State,
                action_courante() : Action,
                actionInitiale() : Action,
                echeanceInitiale() : double,
                estEchue(double, double) : boolean,
                estEffective(double) : boolean,
                estOptimale(Decision) : boolean,
                majEcheanceProchaine(Decision) : double;

procedure       computeInnerState(State)(State),
                computeDecisionState(State)(State, Action),
                ajoutNouvelleRequete(State)(RequestPack),
                ajoutNouvelleManoeuvre(State)(Manoeuvre),
                majCouvertureNuageuse(State)(CloudCoverInformation),
                resetObservation(State)(BasicObservation),
                majEtatDecision(State)(Decision),
                majEtatReelDecision(State)(Decision),
                setDecisionReactive(Decision)(State, Decision, double, Plan),
                amputePlanPremiereAction(Plan)(Decision);

task deliberatif(Decision)(State, Plan, boolean);

%%%=== interface d'entrees/sorties

input   time : double,
        nouvelleRequete : RequestPack,
        nouvelleManoeuvre : Manoeuvre,
        echecObservation : BasicObservation,
        detectionNuage : CloudCoverInformation;

return retourDeliberation;                                    % signal de retour de la tache deliberative

output engagement : Decision;

relation retourDeliberation => time;

%%%=== signaux internes mis en output pour le debug

output  changementEtat,
        horloge : double,
        etat_interne : State,
        plan_courant : Plan,
```

```
        action_courante : Action,
        echeanceProchaine : double,
        echeance,
        decisionReactive : Decision,
        preDecisionReactive : Decision,                                 % decisionReactive a l'instant precedent
        deliberation : Decision;                                        % derniere deliberation


%%%=== corps

%%%=== Controle de la tache deliberative

emit deliberation(DECISION_VIDE);                                       % initialisation
var ech := echeanceInitiale() : double in
        emit echeanceProchaine(ech);
end var;
var plan := initialPlan() : Plan in
        emit plan_courant(plan);
end var;

var     d : Decision,
        c : boolean,
        etat_decision := initialDecisionState() : State in              % on positionne l'etat de decision

        every changementEtat do
                emit deliberation(DECISION_VIDE);
                if estEffective(?echeanceProchaine) then
                        c := true;
                        call computeDecisionState(etat_decision)(?etat_interne, ?action_courante);
                        trap Optimum in
                                loop
                                        exec deliberatif(d)(etat_decision, ?plan_courant, c) return retourDeliberation;
                                        emit deliberation(d);
                                        if(estOptimale(d)) then
                                                exit Optimum;
                                        end if;
                                        c := false
                                end loop
                        end trap
                end if
        end every

end var

||

%%%=== ENGAGEMENT : engagement a chaque echeance

var     d : Decision in

        every echeance do
                call setDecisionReactive(d)(?etat_interne, ?deliberation, ?horloge, ?plan_courant);
                emit decisionReactive(d);
                emit engagement(d);
        end every

end var

||

%%%=== Traitement a chaque prise de decision
var     t : double, e : State, plan := ?plan_courant : Plan in
        loop
                present preDecisionReactive then                                % si une decision vient d'etre prise,
                        call amputePlanPremiereAction(plan)(?preDecisionReactive);
                        emit changementEtat;                                    % on relance le raisonnement
                        t := majEcheanceProchaine(?preDecisionReactive);        % on calcule la prochaine echeance
                        emit echeanceProchaine(t);
                end present;
                pause;
        end loop
end var

||

run PRE [ type Decision / T;
             signal decisionReactive / S,
                    preDecisionReactive / preS ]

||

%%%=== Emission du signal d'echeance

loop
        if estEchue(?echeanceProchaine, ?horloge) then
```

```
                emit echeance;
        end if;
        pause;
end loop

||

%%%=== INTERFACE avec le simulateur du monde reel
var e := initialRealState() : State, e_interne := initialInnerState() : State, a := actionInitiale() : Action in

        loop
                e := etat_reel(?horloge);                                       % on recupere l'etat reel courant
                call computeInnerState(e_interne)(e);                           % on calcule l'etat courant interne
                                                                                % a partir de l'etat reel
                present preDecisionReactive then                                % si une decision a ete prise, on
                                                                                % met a jour l'etat reel courant et
                                                                                % l'etat interne courant
                        call majEtatReelDecision(e)(?preDecisionReactive);
                        call majEtatDecision(e_interne)(?preDecisionReactive);
                end present;

                a := action_courante();                                         % on recupere l'action courante

                present nouvelleRequete then
                        call ajoutNouvelleRequete(e_interne)(?nouvelleRequete);
                        emit changementEtat;
                end present;

                present nouvelleManoeuvre then
                        call ajoutNouvelleManoeuvre(e_interne)(?nouvelleManoeuvre);
                        emit changementEtat;
                end present;

                present echecObservation then
                        call resetObservation(e_interne)(?echecObservation);
                        emit changementEtat;
                end present;

                present detectionNuage then
                        call majCouvertureNuageuse(e_interne)(?detectionNuage);
                        emit changementEtat;
                end present;

                emit etat_interne(e_interne);
                emit action_courante(a);
                pause;
        end loop

end var

||

        %%%=== Horloge interne
        var t0 : double, t : double in
                t0 := initialTime();

                loop
                        t := getTime(t0);
                        emit horloge(t);
                        pause;
                end loop
        end var

end module
```

```
module PRE :

type T;

input S : T;
output preS : T;

loop
   present S then
      var preval := ?S : T in
         pause;
         emit preS(preval)
      end var
   else
      pause
   end present
end loop

end module
```

```
module PRINCIPAL :

%%%=== types

type Action, BasicObservation, CloudCoverInformation, Decision, Manoeuvre, Plan, RequestPack, State;

%%%=== interface d'entrees/sorties

input   time : double;

inputoutput     nouvelleRequete : RequestPack,
                                  nouvelleManoeuvre : Manoeuvre,
                                  echecObservation : BasicObservation,
                                  detectionNuage : CloudCoverInformation,
                                  engagement : Decision;

return retourDeliberation;                                  % signal de retour de la tache deliberative

relation retourDeliberation => time;

%%%=== signaux internes mis en output pour les voir

output  changementEtat,
                      horloge : double,
                      etat_interne : State,
                      plan_courant : Plan,
                      action_courante : Action,
                      echeanceProchaine : double,
                      echeance,
                      decisionReactive : Decision,
                      preDecisionReactive : Decision,       % decisionReactive a l'instant precedent
                      deliberation : Decision;              % derniere deliberation

%%%=== corps

run DECISIONNEL_REACTIF

end module
```

Annexe C

Positionnement des actions de détection parmi les actions d'observation

Nous proposons, ici, un algorithme de positionnement des actions de détection du satellite parmi ses actions d'observation.

1 Description du problème

Définitions

Sur un certain horizon $[t_d; t_f]$ on dispose d'un ensemble $\mathscr{O}$ de zones géographiques à observer. L'objectif est de déterminer les dates optimales (voire des fenêtres temporelles optimales) des actions de détection afin qu'elles recouvrent l'ensemble des zones géographiques.
Une action de détection d est définie par :

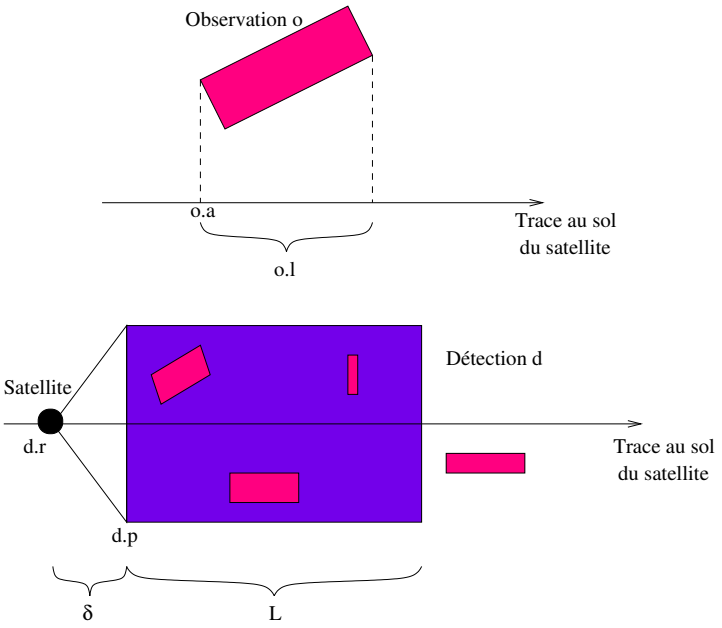
- une abscisse de début de réalisation, $d.r$
- l'abscisse du premier point détecté au sol, $d.p$ ($d.p - d.r = \delta$ constant)
- une longueur de détection au sol, L constante

Une observation o est définie par :

- une abscisse minimale $o.a$
- une longueur (projection de la longueur de la zone sur la trace au sol du satellite) $o.l < L$

Critère

On cherche à minimiser le cardinal de l'ensemble $\mathscr{D}$ des actions de détection.



Contraintes

À la suite d'une détection, le planificateur raisonne sur l'ensemble des zones géographiques entièrement détectées ; cela signifie que la détection partielle d'une zone lui est inutile. On obtient donc la contrainte que toute zone géographique à observer doit être entièrement recouverte par une action de détection :

$$\forall o \in \mathscr{O}, \exists d \in \mathscr{D}, o \subset d$$

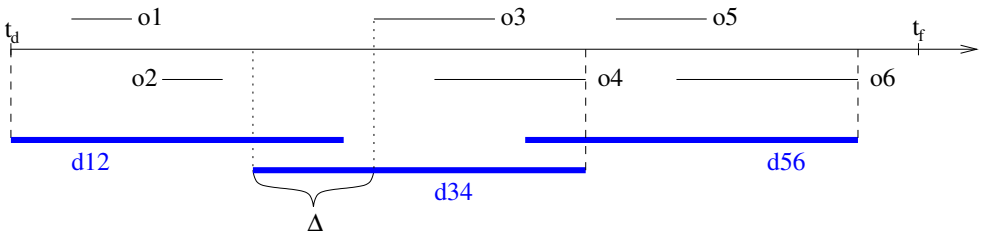


FIG. C.1: Comportement de l'algorithme

2 Résolution

Idée

Intuitivement, il semble intéressant de détecter la couverture nuageuse au plus tôt, pour réserver un maximum de temps à la planification des observations. L'algorithme de calcul des dates optimales de détection consiste donc, à partir de la fin de l'horizon, à recouvrir le plus grand nombre de zones géographiques complètes, toujours à partir de la dernière zone non couverte entièrement. L'algorithme C.1 traduit ce comportement.

Algorithm C.1 Positionnement des actions de détection

Entrées: $\mathscr{O}, t_d, t_f$
Sorties: $\mathscr{D}$

```
𝒟 = ∅
d.r = 0
d.p = δ
d.L = L
tantque 𝒪 ≠ ∅ faire
  α ← max_{o∈𝒪, o.a+o.l≤t_f} (o.a + o.l) // on cherche l'observation la plus lointaine
  d.r = max(α − L − δ, t_d) // la détection doit rester dans l'horizon de raisonnement
  d.p = d.r + δ
  pour o ∈ 𝒪 faire
    // on enlève de 𝒪 toutes les zones recouvertes entièrement par la détection
    si (o.a ≥ d.p) ET (o.a + o.l ≤ d.p + L) alors
      𝒪 ← 𝒪\{o}
    finsi
  fin pour
  𝒟 ← 𝒟 ∪ {d}
fin tantque
retourner 𝒟
```

Optimalité

Nous démontrons ici que cet algorithme est optimal en nombre de détections.

Notations

On note o_i une observation, définie par ses abscisses de début ($o_i.d$) et de fin ($o_i.f$) : $[o_i.d; o_i.f]$.
On note D_i une détection, définie par son abscisse de début ($D_i.d$) et sa longueur constante Δ : $[D_i.d; D_i.d + \Delta]$.

On note $\mathscr{O} = \{o_i\}_{1 \leq i \leq n}$ l'ensemble des observations rangées par ordre d'abscisses de fins *croissantes*. On note enfin $\mathscr{D} = \{D_i\}_{1 \leq i \leq k}$ un ensemble de détections rangées par odre d'abscisses de fins *décroissantes*.
On dit qu'un ensemble de détections $\mathscr{D}$ couvre (ou recouvre) un ensemble d'observations $\mathscr{O}$ si :

$$\forall o \in \mathscr{O}, \exists D \in \mathscr{D} \text{ tel que } o.d \geq D.d \text{ et } o.f \leq D.d + \Delta$$

On note $\mathbb{D}_{\mathscr{O}}$ l'ensemble des ensembles de détections couvrant $\mathscr{O}$.

Démonstration

Notons $\mathscr{D}$ l'ensemble des détections donné par l'algorithme C.1.

$$\mathscr{D} = \{D_1, D_2, ..., D_k\} \in \mathbb{D}_{\mathscr{O}}$$

Nous allons montrer que $\mathscr{D}$ est optimal en nombre de détections.
Remarquons tout d'abord que, par construction, $\mathscr{D}$ recouvre $\mathscr{O}$:

$$\mathscr{D} \in \mathbb{D}_{\mathscr{O}}$$

Reste donc à montrer qu'il existe un ensemble de détections optimal contenant $\mathscr{D}$, soit :

$$\exists \Delta \in \mathbb{D}_{\mathscr{O}} \text{ tel que } \begin{cases} \forall \Delta' \in \mathbb{D}_{\mathscr{O}}, \mathrm{Card}(\Delta') \geq \mathrm{Card}(\Delta) & \textit{(opt. en nb de détections)} \\ \forall i \in [1, k], D_i \in \Delta \end{cases}$$

Cela se fait par récurrence sur i :

$\underline{i = 1}$
L'observation o_n est l'observation dont l'abscisse de fin est la plus grande (o_6 sur la figure C.1).
Par construction, D_1 recouvre o_n.
Supposons que D_1 ne fasse pas partie d'un ensemble optimal ; alors il existerait une détection $D \neq D_1$ telle que :

$$\begin{cases} D \text{ fasse partie d'un ensemble optimal} \\ \text{et } D \text{ recouvre } o_n \end{cases}$$

Comme $D \neq D_1$ alors D_1 débute avant D, et couvre donc entièrement au moins autant d'observations que D.

Il existe donc un ensemble de détections optimal contenant D_1

$\underline{i = 2}$
Soit $1 \leq m \leq n$ le plus grand entier tel que o_m ne soit pas entièrement couverte par D_1 (o_4 sur la figure C.1).
Par construction, D_2 recouvre o_m.
Supposons que D_2 ne fasse pas partie d'un ensemble optimal contenant D_1 ; alors il existerait une détection $D \neq D_2$ telle que :

$$\begin{cases} D \text{ fasse partie d'un ensemble optimal contenant } D_1 \\ \text{et } D \text{ recouvre } o_m \end{cases}$$

Comme $D \neq D_2$ alors D_2 débute avant D, et couvre donc entièrement au moins autant d'observations que D.

Il existe donc un ensemble de détections optimal contenant D_1 et D_2

Hypothèse de récurrence : Il existe un ensemble de détections optimal contenant D_1, D_2, ..., D_i.

$\underline{i \rightarrow i + 1}$
L'hypothèse de récurrence est vérifiée aux rangs 1 et 2. Supposons-la vraie au rang i et démontrons-la au rang $i + 1$.

Soit $1 \leq p \leq n$ le plus grand entier tel que o_p ne soit pas entièrement couverte par D_i.
Par construction, D_{i+1} recouvre o_p.
Supposons que D_{i+1} ne fasse pas partie d'un ensemble optimal contenant D_1, D_2, ..., D_i ; alors il existerait une détection $D \neq D_{i+1}$ telle que :

$$\begin{cases} D \text{ fasse partie d'un ensemble optimal contenant } D_1, D_2, ..., D_i \\ \text{et } D \text{ recouvre } o_p \end{cases}$$

Comme $D \neq D_{i+1}$ alors D_{i+1} débute avant D, et couvre donc entièrement au moins autant d'observations que D.

Il existe donc un ensemble de détections optimal contenant D_1, D_2, ..., D_i, D_{i+1}

Conclusion : Nous avons montré par récurrence qu'il existe un ensemble de détections optimal, $\mathscr{D}^*$, contenant $\{D_1, D_2, ..., D_k\} = \mathscr{D}$.

$\mathscr{D}$ est donc un ensemble de détections optimal en nombre de détections

Amélioration

Il n'est pas nécessaire de calculer les dates précises de réalisation des actions de détection ; parfois un intervalle temporel suffit et permet d'assouplir la planification (voir algorithme C.2).

Algorithm C.2 Fenêtres de détection

Entrées: $\mathscr{O}, t_d, t_f$
Sorties: $\mathscr{D}$

$\mathscr{D} = \emptyset$
$d.r = 0$
$d.p = \delta$
$d.L = L$
tantque $\mathscr{O} \neq \emptyset$ **faire**
  $\alpha \leftarrow \max_{o \in \mathscr{O}, o.a + o.l \leq t_f} (o.a + o.l)$ // on cherche l'observation la plus lointaine
  $d.r = \max(\alpha - L - \delta, t_d)$ // la détection doit rester dans l'horizon de raisonnement
  $d.p = d.r + \delta$
  $\Delta \leftarrow L$
  pour $o \in \mathscr{O}$ **faire**
    // on enlève de $\mathscr{O}$ toutes les zones recouvertes entièrement par la détection
    si $(o.a \geq d.p)$ ET $(o.a + o.l \leq d.p + L)$ **alors**
      $\mathscr{O} \leftarrow \mathscr{O} \backslash \{o\}$
      $\Delta = \min(o.a - d.p, \Delta)$
    finsi
  fin pour
  $d.r = [d.r; d.r + \Delta]$ // relâchement des contraintes sur $d.r$
  $d.p = d.r + \delta$
  $\mathscr{D} \leftarrow \mathscr{D} \cup \{d\}$
fin tantque
retourner $\mathscr{D}$

Réduction de $\mathscr{O}$

Il existe certaines zones sur la Terre où les données climatologiques sont plus fiables que d'autres : proche des tropiques par exemple il y a une très faible probabilité de couverture nuageuse. On peut donc éviter de placer des actions de détection à ces endroits.

Pour cela nous avons besoin de savoir à partir de quelle probabilité de couverture nuageuse prévue par les données climatologiques il est préférable que le satellite détecte lui-même la couverture nuageuse.

On note :

o une zone au sol à observer,

p_o la probabilité, donnée par les prévisions climatologiques, qu'il y ait des nuages au dessus de o,

c_o le coût de l'observation de o (perte de temps, consommation d'énergie, ...),

c_d le coût de la détection de la couche nuageuse au dessus de o,

g_o le gain associé à la réalisation élémentaire de o.

On évalue le gain que le satellite retirerait dans le cas de la réalisation de l'observation o

– si on ne détecte pas la couverture nuageuse à bord

$$gain_{\neg d} = \max\left(-c_o + (1-p_o)g_o, 0\right)$$

– si on détecte la couverture nuageuse à bord

$$gain_d = -c_d + (1-p_o)(g_o - c_o)$$

On en déduit donc une condition nécessaire et suffisante pour la réalisation d'une action de détection :

$$\begin{aligned}
\text{détection ssi} \quad & gain_d > gain_{\neg d} \\
\text{ssi} \quad & -c_d + (1-p_o)(g_o - c_o) > \max\left(-c_o + (1-p_o)g_o, 0\right) \\
\text{ssi} \quad & \begin{cases} -c_d + (1-p_o)(g_o - c_o) > -c_o + (1-p_o)g_o \\ -c_d + (1-p_o)(g_o - c_o) > 0 \end{cases} \\
\text{détection ssi} \quad & \begin{cases} -c_d + (1-p_o)(g_o - c_o) > -c_o + (1-p_o)g_o \\ p_o c_o > c_d \end{cases}
\end{aligned}$$

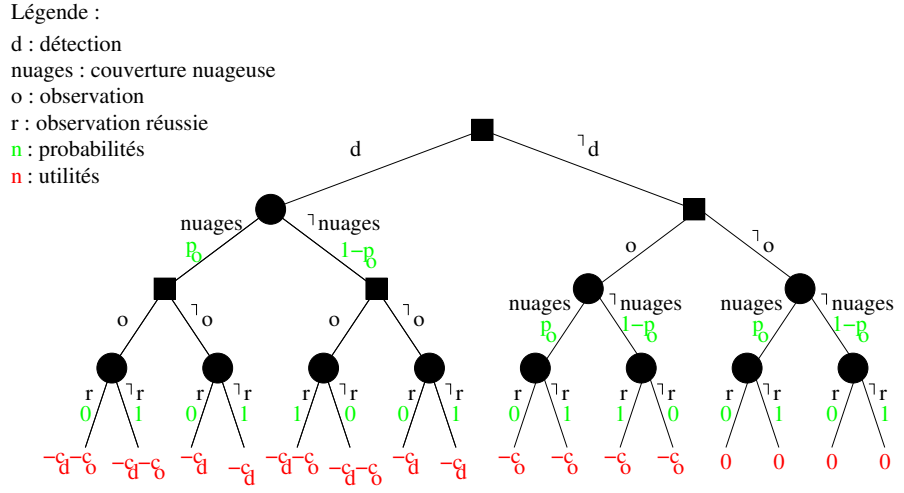


FIG. C.2: Arbre de décision

Annexe D

Code source de l'algorithme glouton stochastique itéré

On donne ici le code source de quelques méthodes des classes `StochasticGreedyPlannerDR1` et `StochasticDecisionRule1` correspondant respectivement à l'algorithme glouton stochastique itéré et à la règle de décision 1 bruitée (SDR1).

1 Descente stochastique dans l'arbre de recherche

```
/**
 * Methode de la classe StochasticGreedyPlannerDR1 qui effectue une
 * descente aleatoire dans l'arbre de recherche. Cette descente
 * s'effectue a partir de l'etat de decision et jusqu'a la fin de
 * l'horizon de planification.
 */
public void plan() {
  this.ok = true;
  double startTime = this.decisionState.getTime();
  double currentTime = startTime;
  // On ne fait qu'appliquer la regle de de decision 1 bruitee a chaque prise de decision
  StochasticDecisionRule1 stochasticDR1 = new StochasticDecisionRule1(this.decisionState, this.decisionHorizon, this.
       model, this.priorityProbability, this.observationProbability);
  ArrayList<Action> actions = new ArrayList<Action>(); // Action, ou couple d'actions "changement d'attitude + action"
       decide
  State newState;
  do {
    // Decision de la regle de decision bruitee
    actions = stochasticDR1.getDecision();
    for (int i = 0; i < actions.size(); i++) {
      // On simule l'execution de l'action
      actions.get(i).execution(stochasticDR1.getDecisionState());
      // On ajoute l'action a la suite du plan
      this.plan.addAction(actions.get(i));
      // On redefinit le nouvel etat de decision comme l'etat de fin de l'action qui vient d'etre ajoutee
      newState = actions.get(i).getEndState();
      stochasticDR1.setDecisionState(newState.clone());
    }
    currentTime = stochasticDR1.getDecisionState().getTime();
  } while ((!actions.isEmpty()) && (currentTime < startTime + this.planningHorizon) && (currentTime < this.model.
       getScenarioEndTime()));
}
```

2 Prise de décision bruitée

```
/**
 * Methode de la classe StochasticDecisionRule1 qui renvoie l'action
 * decidee, ou le couple d'actions "changement d'attitude + action"
 * si un changement d'attitude est necessaire, en suivant la regle de
 * decision 1 bruitee.
 */
public ArrayList<Action> getDecision() {
  ArrayList<Action> result = new ArrayList<Action>();
  Action mainAction = new Action();

  // Une manoeuvre orbitale est-elle possible ?
  this.orbitalManoeuvresSet = this.decisionState.getOrbitalManoeuvreSet(this.model, this.decisionHorizon);
  if (!this.orbitalManoeuvresSet.isEmpty() && (Math.random() <= this.priorityProbability)) {
    mainAction = this.orbitalManoeuvresSet.get(0);
    double earliestStartTime = this.orbitalManoeuvresSet.get(0).getStartTime();
    // On recupere la manoeuvre orbitale la plus proche de l'instant de decision
    for (OrbitalManoeuvre man : this.orbitalManoeuvresSet) {
      if (man.getStartTime() < earliestStartTime) {
        mainAction = man;
        earliestStartTime = man.getStartTime();
      }
    }
  } else {
    // Un teledechargement est-il possible ?
    this.downloadsSet = this.decisionState.getDownloadSet(this.model,this.decisionHorizon);
    if (!this.downloadsSet.isEmpty() && (Math.random() <= this.priorityProbability)) {
      if (this.downloadsSet.size() == 1) {
      // S'il n'y a qu'un seul teledechargement, on le choisit
        mainAction = this.downloadsSet.get(0);
      } else {
        // Sinon, on choisit le teledechargement qui contient l'observation elementaire de plus grande priorite
        int maxPriorityDownload = -1;
        for (Download d : this.downloadsSet) {
          // On recupere la priorite max des observations a teledecharger
          if (d.getBasicObservationMaxPriority() > maxPriorityDownload) {
            maxPriorityDownload = d.getBasicObservationMaxPriority();
          }
        }
        ArrayList<Download> maxPriorityDownloadList = new ArrayList<Download>();
        for (Download d : this.downloadsSet) {
          // On construit la liste des teledechargements dont la priorite max des observations elementaires est la
              priorite max
          if (d.getBasicObservationMaxPriority() == maxPriorityDownload) {
            maxPriorityDownloadList.add(d);
          }
        }
        if (maxPriorityDownloadList.size() == 1) {
          // S'il n'y a qu'un seul teledechargement, on le choisit
          mainAction = maxPriorityDownloadList.get(0);
        } else {
          // Sinon, on choisit le teledechargement le plus proche dans le tems de l'etat de decision
          double earliestStartTime = this.decisionState.getTime()  + this.decisionHorizon + 1;
          for (Download d : maxPriorityDownloadList) {
            // On recupere la date de debut au plus tot d'un teledechargement
            if (d.getEarliestStartTime(this.decisionState.getTime()) < earliestStartTime) {
              earliestStartTime = d.getEarliestStartTime(this.decisionState.getTime());
            }
          }
          ArrayList<Download> maxPriorityEarliestDownloadList = new ArrayList<Download>();
          for (Download d : maxPriorityDownloadList) {
            // On construit la liste des teledechargements de priorite max et de date de debut au plus tot minimale
            if (d.getEarliestStartTime(this.decisionState.getTime()) == earliestStartTime) {
              maxPriorityEarliestDownloadList.add(d);
            }
          }
          if (maxPriorityEarliestDownloadList.size() == 1) {
            // S'il n'y a plus qu'un seul teledechargement, on le choisit
            mainAction = maxPriorityEarliestDownloadList.get(0);
          } else {
            // Sinon, on choisit le teledechargement vers le centre de mission dont le nom est le premier dans l'ordre
                alphabetique
            String maxName = "zzzzzz";
            for (Download d : maxPriorityEarliestDownloadList) {
              if (d.getMissionCenter().getName().compareTo(maxName) < 0) {
                mainAction = d;
                maxName = d.getMissionCenter().getName();
              }
            }
          }
        }
      }
```

```
    } else {
      // Une detection est-elle possible ?
      this.detectionsSet = this.decisionState.getDetectionSet(this.model, this.decisionHorizon);
      if (!this.detectionsSet.isEmpty() && (Math.random() <= this.priorityProbability)) {
        mainAction = this.detectionsSet.get(0);
      } else {
        // Une observation est-elle possible ? On ne choisit pas une observation si on est en eclipse
        this.observationsSet = this.decisionState.getObservationSet(this.model,this.decisionHorizon);
        if (!this.observationsSet.isEmpty() && (Math.random() <= this.priorityProbability)) {
          if (Math.random() > this.observationProbability) {
            // On choisit une observation au hasard
            Random random = new Random();
            mainAction = this.observationsSet.get(random.nextInt(this.observationsSet.size()));
          } else {
            // On choisit l'observation la plus prioritaire
            double maxPriority = -1;
            for (Observation o : this.observationsSet) {
              // On recupere la priorite max
              if (o.getBasicObservation().getComplexObservation().getPriority() > maxPriority) {
                maxPriority = o.getBasicObservation().getComplexObservation().getPriority();
              }
            }
            ArrayList<Observation> maxPriorityList = new ArrayList<Observation>();
            for (Observation o : this.observationsSet) {
              // On construit la liste des observations de priorite max
              if (maxPriority == o.getBasicObservation().getComplexObservation().getPriority()) {
                maxPriorityList.add(o);
              }
            }
            if (maxPriorityList.size() == 1) {
              // S'il n'y a plus qu'une seule observation, on la choisit
              mainAction = maxPriorityList.get(0);
            } else {
              // Sinon, soit on choisit l'observation la plus proche dans le temps de l'etat de decision (avec une
                   probabilite de 0.5)
              if (Math.random() < 0.5) {
                double earliestStartTime = this.decisionState.getTime()  + this.decisionHorizon + 1;
                for (Observation o : maxPriorityList) {
                  // On recupere la date de debut au plus tot d'une observation
                  if (o.getEarliestStartTime(this.decisionState.getTime()) < earliestStartTime) {
                    earliestStartTime = o.getEarliestStartTime(this.decisionState.getTime());
                  }
                }
                ArrayList<Observation> maxPriorityEarliestList = new ArrayList<Observation>();
                for (Observation o : maxPriorityList) {
                  // On construit la liste des observations de priorite max et de date de debut au plus tot minimale
                  if (o.getEarliestStartTime(this.decisionState.getTime()) == earliestStartTime) {
                    maxPriorityEarliestList.add(o);
                  }
                }
                if (maxPriorityEarliestList.size() == 1) {
                  // S'il n'y a plus qu'une seule observation, on la choisit
                  mainAction = maxPriorityEarliestList.get(0);
                } else {
                  // Sinon, on choisit l'observation de plus petit index
                  long minIndex = Long.MAX_VALUE;
                  for (Observation o : maxPriorityEarliestList) {
                    if (o.getBasicObservation().getIndex() < minIndex) {
                      mainAction = o;
                      minIndex = o.getBasicObservation().getIndex();
                    }
                  }
                }
              } else {
                // Soit on choisit l'observation la moins nuageuse
                double lowestCloudCover = 1;
                for (Observation o : maxPriorityList) {
                  // On recupere la couverture nuageuse la plus faible
                  if (o.getBasicObservation().getState(this.decisionState.getTime(),this.model.getCloudCover()).
                       getCloudCover() < lowestCloudCover) {
                    lowestCloudCover = o.getBasicObservation().getState(this.decisionState.getTime(),this.model.
                         getCloudCover()).getCloudCover();
                  }
                }
                ArrayList<Observation> maxPriorityLowestList = new ArrayList<Observation>();
                for (Observation o : maxPriorityList) {
                  // On construit la liste des observations de priorite max et de couverture nuageuse minimale
                  if (o.getBasicObservation().getState(this.decisionState.getTime(),this.model.getCloudCover()).
                       getCloudCover() == lowestCloudCover) {
                    maxPriorityLowestList.add(o);
                  }
                }
                if (maxPriorityLowestList.size() == 1) {
                  // S'il n'y a plus qu'une seule observation, on la choisit
                  mainAction = maxPriorityLowestList.get(0);
```

```
              } else {
                // Sinon, on choisit l'observation de plus petit index
                long minIndex = Long.MAX_VALUE;
                for (Observation o : maxPriorityLowestList) {
                  if (o.getBasicObservation().getIndex() < minIndex) {
                    mainAction = o;
                    minIndex = o.getBasicObservation().getIndex();
                  }
                }
              }
            }
          }
        } else {
          // Un pointage Soleil est-il possible ? On ne choisit pas un pointage Soleil si on est en eclipse
          this.sunPointingsSet = this.decisionState.getSunPointingSet(this.model,this.decisionHorizon);
          if (!this.sunPointingsSet.isEmpty() && (Math.random() <= this.priorityProbability)) {
            mainAction = this.sunPointingsSet.get(0);
          } else {
            // Un pointage geocentrique est-il possible ?
            this.geoPointingsSet = this.decisionState.getGeoPointingSet(this.model,this.decisionHorizon);
            if (!this.geoPointingsSet.isEmpty()) {
              mainAction = this.geoPointingsSet.get(0);
            }
          }
        }
      }
    }
  }

  mainAction.execution(this.decisionState.clone());

  AttitudeChange newAttitudeChange = null;
  if (!mainAction.isPossible()) {
    newAttitudeChange = mainAction.neededAttitudeChange(this.decisionState, this.model.getSatellite());
    if (newAttitudeChange != null) {
      newAttitudeChange.setName(newAttitudeChange.getName() + " (to reach " + mainAction.getName() + ")");
      newAttitudeChange.execution(this.decisionState);
      result.add(newAttitudeChange);
      mainAction.getStartState().maj(newAttitudeChange.getEndState());

    }
  }
  result.add(mainAction);

  return result;
}
```

Bibliographie

E. Aarts et J. Lenstra. *Local Search in Combinatorial Optimization*. John Wiley & Sons, 1997.

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

M.-L. Anadon, M. Fuveau, L. Kerjean, B. Laborde, B. Masson, J.-M. Mesnager, Y. Mulet, X. Passot, R. Soumagne, et C. Villaret. *Guide de spécification du Commande-Contrôle*. 2005.

C. Barrouil et J. Lemaire. Advanced real-time mission management for an AUV. Dans *Proceedings of the SCI NATO Symposium on Advanced Mission Management and System Integration Technologies for Improved Tactical Operations*, Florence, Italie, Septembre 1999.

G. Beaumet. Continuous planning for the control of an autonomous agile satellite. Dans *Doctoral Consortium of the International Conference on Automated Planning and Scheduling (ICAPS'06)*, Ambleside, Royaume-Uni, 6-10 Juin 2006.

G. Beaumet, G. Verfaillie, et M.-C. Charmeau. Planification continue pour la conduite d'un satellite agile autonome. Dans *Journées Francophones Planification, Décision, Apprentissage pour la conduite de systèmes (JFPDA'06)*, Toulouse, France, 10-12 Mai 2006.

G. Beaumet, G. Verfaillie, et M.-C. Charmeau. Estimation of the minimal duration of an attitude change for an autonomous agile earth-observing satellite. Dans *International Conference on Principles and Practice of Constraint Programming (CP'07)*, Providence, RI, États-Unis, 23-27 Septembre 2007a.

G. Beaumet, G. Verfaillie, et M.-C. Charmeau. Estimation des durées minimales de basculement pour un satellite d'observation agile autonome. Dans *Journées Francophones de Programmation par Contraintes (JFPC'07)*, Rocquencourt, France, 4-6 Juin 2007b.

G. Beaumet, G. Verfaillie, et M.-C. Charmeau. Autonomous planning for an agile earth-observing satellite. Dans *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS'08)*, Los Angeles, CA, États-Unis, 25-29 Février 2008a.

G. Beaumet, G. Verfaillie, et M.-C. Charmeau. Decision-making on-board an autonomous agile earth-observing satellite. Dans *International Conference on Automated Planning and Scheduling (ICAPS'08) Workshop on Scheduling and Planning Applications*, Sydney, Australie, 14-18 Septembre 2008b.

G. Berry et G. Gonthier. The ESTEREL synchronous programming language : Design, semantics, implementation. *Science of Computer Programming*, 19(2) :87–152, 1992.

D. Bertsekas et J. Tsitsiklis. *Neuro Dynamic Programming*. Athena Scientific, 1996.

A. Blum et M. Furst. Fast planning throught planning graph analysis. Dans *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1636–1642, Montréal, Canada, 1995.

R. P. Bonasso, J. Firby, E. Gat, D. Kortenkamp, D. P. Miller, et M. G. Slack. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental & Theoretical Artificial Intelligence*, 9(2/3) :237–256, Avril 1997.

E. Boussarie et B. Boissin. PLÉIADES - Système dual d'observation optique de résolution métrique. `http://smsc.cnes.fr/PLEIADES/`, 2006.

J. Bresina. Heuristic-biased stochastic sampling. Dans *National Conference on Artificial Intelligence (AAAI'96)*, pages 271–278, Portland, OR, États-Unis, 1996.

R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1) :14–23, Mars 1986.

J. Busseuil, M. Llibre, et X. Roser. High precision mini-cmg's and their spacecraft applications. Dans *Proceedings of the Annual AAS Rocky Mountain Guidance and Control Conference*, Breckenridge, Colorado, États-Unis, 4-8 Février 1998.

E. Chanthery. *Planification de mission pour un véhicule aérien autonome*. Thèse de doctorat, Supaéro, Toulouse, France, 2005.

M.-C. Charmeau et E. Bensana. AGATA, a lab bench project for spacecraft autonomy. Dans *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS'08)*, Los Angeles, CA, États-Unis, 25-29 Février 2008.

S. Chien, R. Knight, A. Stechert, R. Sherwood, et G. Rabideau. Using iterative repair to improve the responsiveness of planning and scheduling. Dans *Proceedings of the 5th*

International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00), pages 300–307, Breckenridge, CO, États-Unis, 2000a.

S. Chien, G. Rabideau, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, et D. Tran. ASPEN : automated planning and scheduling for space mission operations. Dans *Proceedings of the 6th International Symposium on Space Operations at the Start of the 3rd Millennium (SpaceOps'00)*, Toulouse, France, 2000b.

S. Chien, R. Sherwood, D. Tran, et B. Cichy. The EO-1 Autonomous Science Agent. Dans *the international conference on Autonomous Agents and Multi-Agent Systems (AAMAS'04)*, pages 420–427, New York, NY, États-Unis, 2004.

V. Cicirello et S. Smith. Enhancing stochastic search performance by value-biased randomization of heuristics. *Journal of Heuristics*, 11(1) :5–34, 2005.

CNES. *Techniques & Technologies des Véhicules Spatiaux*, chapitre Mécanique spatiale, pages 79–311. Cépaduès, 1998.

S. Damiani. *Gestion d'une constellation de satellites de surveillance de la Terre : autonomie et coordination.* Thèse de doctorat, Supaéro, Toulouse, France, 2005.

D. Escorial, I. Tourne, et F. Reina. FUEGO : a dedicated constellation of small satellites to detect and monitor forest fires. Dans *Third IAA Symposium on Small Satellites for Earth Observation*, Berlin, Allemagne, 2-6 Avril 2001.

T. Estlin, R. Volpe, I.A.D Nesnas, D. Mutz, F. Fisher, B. Engelhardt, et S. Chien. Decision-making in a robotic architecture for autonomy. Dans *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS'01)*, Montréal, Canada, 18-21 Juin 2001.

C. Fargeon. *Robotique mobile*, chapitre Introduction : Émergence d'une science, pages 1–9. Teknea, 1993.

R.J. Firby. An investigation into reactive planning in complex domains. Dans *National Conference on Artificial Intelligence (AAAI'87)*, pages 202–206, Seattle, WA, États-Unis, 1987.

S. Fleury, M. Herbb, et R. Chatila. Design of a modular architecture for autonomous robot. Dans *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'94)*, pages 3508–3513, San Diego, CA, États-Unis, 1994.

E. Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. Dans *National Conference on Artificial Intelligence (AAAI'92)*, pages 809–815, San Jose, CA, États-Unis, 1992.

E. Gat. ESL : A language for supporting robust plan execution in embedded autonomous agents. Dans *Proceedings of the IEEE Aerospace Conference*, volume 1, pages 319–324, Aspen, CO, États-Unis, Février 1997.

A. Gerevini et D. Long. Preferences and soft constraints in PDDL3. Dans *International Conference on Automated Planning and Scheduling (ICAPS'06) Workshop on Preferences and Soft Constraints in Planning*, pages 46–53, Ambleside, Royaume-Uni, 6-10 Juin 2006.

M. Ghallab et H. Laruelle. Representation and control in IxTeT, a temporal planner. Dans *Proceedings of the Second International Conference on AI Planning Systems (AIPS'94)*, pages 61–67, Chicago, IL, États-Unis, 13-15 Juin 1994.

M. Ghallab, D. Nau, et P. Traverso. *Automated Planning*. Morgan Kaufmann, 2004.

L. Granvilliers et F. Benhamou. *Rossi, F., Beek, P.V., Walsh, T., eds. : Handbook of Constraint Programming.*, chapitre Continuous and Interval Constraints, pages 571–603. Elsevier, 2006a.

L. Granvilliers et F. Benhamou. Realpaver : an interval solver using constraint satisfaction techniques. *ACM Trans. Math. Softw.*, 32(1) :138–156, 2006b. ISSN 0098-3500. doi : http://doi.acm.org/10.1145/1132973.1132980.

J. Guitton, J.-L. Farges, et R. Chatila. A planning architecture for mobile robotics. Dans *Proceedings of the 1st Mediterranean Conference on Intelligent Systems and Automation*, Annaba, Algérie, Juin 2008.

Keith Halsey, Derek Long, et Maria Fox. CRIKEY - A temporal planner looking at the integration of scheduling and planning. Dans *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS'04) Workshop on Integrating Planning into Scheduling*, Whistler, British Columbia, Canada, 4 Juin 2004.

K. J. Hammond. Explaining and repairing plans that fail. *Artificial Intelligence*, 45 : 173–228, 1990.

M. Hassoun. *Contrôle d'exécution des mouvements d'un robot mobile : application à l'assistance à la conduite automobile*. Thèse de doctorat, Institut National Polytechnique de Grenoble, Laboratoire d'Informatique Fondamentale et d'Intelligence Artificielle, Grenoble, France, 1997.

S. Kambhampati et J. A. Hendler. Control of refitting during plan reuse. Dans *Proceedings International Joint Conference on Artificial Intelligence (IJCAI'89)*, pages 943–948, Detroit, MI, États-Unis, 1989a. Morgan Kaufmann.

S. Kambhampati et J. A. Hendler. Flexible reuse of plans via annotation and verification. Dans *Fifth Conference on Artificial Intelligence Applications*, pages 37–43, Miami, FL, États-Unis, 1989b. IEEE.

S. Kambhampati et J. A. Hendler. A validation structure based theory of plan modification and reuse. *Artificial Intelligence*, 55(2/3) :193–258, 1992.

L. Khatib, J. Frank, D. Smith, R. Morris, et J. Dungan. Interleaved observation execution and rescheduling on earth observing systems. Dans *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS'03) Workshop on Plan Execution*, Trento, Italie, 9-13 Juin 2003.

J. Koehler. Planning under resource constraints. Dans *Proceedings of the 13th European Conference on Artificial Intelligence*, pages 489–493, Brighton, Royaume-Uni, 1998.

J.-M. Lachiver, J.-M. Laherrère, I. Sebbag, N. Bataille, et T. Bert-Dibat. System feasibility of on-board clouds detection and observations scheduling. Dans *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS'01)*, Montréal, Canada, 18-22 Juin 2001.

H. Laruelle, P. Régnier, et S. Thiébaux. Dossier "planification et action". Rapport technique 94/1/R, Institut de Recherche en Informatique de Toulouse, 1994.

J. Latombe. *Robot Motion Planning*. Kluwer Academic, 1991.

Steven M. LaValle. *Planning algorithms*, chapitre Sampling-based planning under differential constraints, pages 707–707. Cambridge University Press, 2006.

M. Lemaître. Interaction between reactive and deliberative tasks for on-line decision-making. Dans *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS'07) Workshop on Planning and Plan Execution for Real-World Systems*, Providence, RI, États-Unis, 2007.

M. Lemaître et G. Verfaillie. Interaction entre tâches réactives et délibératives pour la décision en ligne. Dans *Journées Francophones Planification, Décision, Apprentissage pour la conduite de systèmes (JFPDA'07)*, Grenoble, France, 2-6 Juillet 2007.

M. Llibre. Agata (phase 3) - guidage autonome. prise en compte des gyrodynes dans la simulation des prises de vue. Rapport technique RI 4/12094 DCSD, ONERA, Juin 2007.

G. Margulies et J.N. Aubrun. Geometric theory of single-gimbal control moments gyro systems. *Journal of the Astronauticals Sciences*, XXVI(2) :159–191, Avril-Juin 1978.

J.P. Merlet. Interval Analysis and Robotics (Invited Presentation). Dans *Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP'06)*, 24-29 Septembre 2006.

S. Minton, J. Bresina, et M. Drummond. Total-order and partial-order planning : A comparative analysis. *Journal of Artificial Intelligence Research*, 2 :227–262, 1994.

U. Montanari. Networks of constraints : Fundamental properties and applications to picture processing. *Information Sciences*, 7(2) :95–132, 1974.

N. Muscettola. Hsts : Integrated planning and scheduling. Dans *Fox, M., and Zweben, M., eds, Intelligent Scheduling*, pages 169–212. Morgan Kaufman, 1994.

N. Muscettola, S. Smith, A. Cesta, et D. D'Aloisi. Coordinating space telescope operations in an integrated planning and scheduling architecture. *IEEE Control Systems*, 12(1) : 28–37, 1992.

N. Muscettola, P. Nayak, B. Pell, et B. Williams. Remote agent : To boldly go where no AI system has gone before. *Artificial Intelligence*, 103 :5–47, 1998.

N. Muscettola, G. Dorais, C. Fry, R. Levinson, et C. Plaunt. IDEA : Planning at the core of autonomous reactive agents. Dans *Proceedings of the third NASA International Workshop on Planning and Scheduling for Space*, Houston, TX, États-Unis, Octobre 2002.

P. Parraud, A. Flipo, J. Jaubert, et G. Lassalle-Balier. Computing smooth attitude guidance laws for homing maneuvers. Dans *International Symposium on Space Technology and Science*, Kanazawa, Japan, 2006.

E. Pednault. ADL : Exploring the middle ground between strips and the situation calculus. Dans *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 324–332, Toronto, Canada, 15-18 Mai 1989.

B. Pell, E. Gat, R. Keesing, N. Muscettola, et B. Smith. Robust periodic planning and execution for autonomous spacecraft. Dans *International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 1234–1239, Nagoya, Japan, 23-29 Août 1997.

B. Pell, E. Gamble, E. Gat, R. Keesing, J. Kurien, W. Millar, P. Nayak, C. Plaunt, et B. Williams. A hybrid procedural/deductive executive for autonomous spacecraft. Dans *International Conference on Autonomous Agents*, pages 369–376, Minneapolis, MN, États-Unis, 10-13 Mai 1998.

M. Puterman. *Markov Decision Processes, Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

F. Rossi, P. van Beek, et T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

R. G. Simmons. Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1) :34–43, 1994.

D. Smith et D. Weld. Temporal planning with mutual exclusion reasoning. Dans *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 326–337, Stockholm, Suède, 31 Juillet - 6 Août 1999.

D. Smith, J. Frank, et A. Jónsson. Bridging the gap between planning and scheduling. *Knowledge Engineering Review*, 15(1) :47–83, 2000.

G. Verfaillie. Synthèse sur les besoins en autonomie des systèmes spatiaux. Rapport technique, LAAS-CNRS, Groupe RIA, Toulouse, France, 2005.

G. Verfaillie. Architecture générique de contrôle d'un engin spatial autonome. Rapport technique, ONERA, Toulouse, France, 2006.

G. Verfaillie, C. Pralet, et M. Lemaître. Constraint-based modeling of discrete event dynamic systems. *Journal of Intelligent Manufacturing, Special Issue on "Planning, Scheduling, and Constraint Satisfaction"*, 2009.

B. Williams et P. Nayak. A model-based approach for reactive self-configuring systems. Dans *National Conference on Artificial Intelligence (AAAI'96)*, volume 2, pages 971–978, Portland, OR, États-Unis, 4-8 Août 1996.

L. Zaffalon. *Programmation synchrone de systèmes réactifs avec Esterel et les SyncCharts*. Presses Polytechniques et Universitaires Romandes, 2005.

S. Zilberstein. Using anytime algorithms in intelligent systems. Dans *AI Magazine*, volume 17, pages 73–83, 1996.