Institut National Polytechnique de Toulouse

ENSEEIHT **École doctorale EDSYS**

Sélection et contrôle de modes de déplacement pour un robot mobile autonome en environnements naturels

THÈSE

présentée et soutenue publiquement le 18 juillet 2006

pour l'obtention du

Doctorat de l'Institut National Polytechnique de Toulouse

(spécialité systèmes automatiques)

par

Thierry Peynot

Composition du jury

Rapporteurs :	Philippe Bidaud	Professeur, Université Paris 6
	Bernard Dubuisson	Professeur, Université de Technologie de Compiègne
Examinateurs :	Catherine Tessier	Ingénieur de recherche, ONERA-CERT
	Ronald C. Arkin	Professeur, Georgia Institute of Technology, USA
	Simon Lacroix	Chargé de Recherche, LAAS-CNRS
Directeur de thèse :	Raja Chatila	Directeur de Recherche, LAAS-CNRS

Laboratoire d'Analyse et d'Architecture des Systèmes — UPR CNRS

Remerciements

Après quelques quatre années de recherche semées d'embûches, le moment d'écrire ces quelques lignes arrive un peu comme une "libération", et revêt un caractère d'une importance toute particulière à mes yeux, comme bien souvent aux yeux du lecteur.
Tout d'abord car cette étape est généralement effectuée une fois la thèse rédigée et soutenue, donc pour ainsi dire terminée. Cela signifie que "le plus dur est fait", avec une conclusion (malgré tout) généralement plutôt positive, et permet une rédaction dans une atmosphère plus détendue que lors des dernières semaines de la rédaction (en particulier...).
D'autre part car il s'agit finalement de la partie la plus "personnelle" du mémoire de thèse, celle qui peut donner une indication sur la personnalité de son auteur, les conditions dans lesquelles la thèse a été effectuée, encadrée (si tel était le cas...), et soutenue par un entourage.

On dit souvent que malgré la présence d'un unique nom dans le champ *auteur* du document, un tel travail n'a pas été réalisé seul, et n'aurait pu exister sans l'aide, le soutien et la contribution d'autres personnes. Pour quiconque ayant déjà affronté cette "épreuve", cela peut sembler une lapalissade, mais il est des vérités qui sont toujours bonnes à dire et même à répéter. De plus, ce ne sont pas forcément les personnes les plus "évidentes" qui ont finalement apporté les contributions et supports les plus fondamentaux.

Je souhaite ainsi remercier de nombreuses personnes que j'ai été amené à côtoyer pendant ces quelques années de thèse, que ce soit dans un cadre de travail ou plus amical, en espérant n'oublier personne. Il est entendu que le lecteur ne devra lier aucune échelle "d'importance" à l'ordre dans lequel apparaîtront ces divers noms. Il serait de toutes manières bien dérisoire de vouloir seulement en attribuer un.

Pour commencer, merci à Malik Ghallab, directeur du LAAS-CNRS pour m'avoir permis de réaliser ces travaux dans ce laboratoire, et à Raja Chatila, responsable du groupe RIA[1], pour avoir accepté d'être mon directeur de thèse "officiel".

Merci à Simon Lacroix, encadrant de cette thèse, pour sa confiance et la liberté dont j'ai bénéficié pour mener ces travaux de recherche. Son enthousiasme et sa motivation pour "faire bouger les robots" est toujours communicative, même en ne le voyant qu'entre deux portes.

Merci également aux deux rapporteurs Philippe Bidaud (LRP) et Bernard Dubuisson (UTC), et aux examinateurs Catherine Tessier (ONERA) et Ronald C. Arkin (Georgia Tech, USA) d'avoir accepté de participer au jury de ma thèse.

Tous mes remerciements à Matthieu et Sara, qui font "tourner la boutique" et rendent possible le côté expérimental sur les robots qui fait (à mon avis) la beauté et la source de motivation principale de notre travail. Merci aussi en général aux services techniques du LAAS, dont Sysadmin et les électroniciens.

Un grand merci à Maître Patrick D., dont la modestie n'a d'égale que la compétence et la remarquable (et rare) rigueur scientifique. Nos quelques brainstorming auront en particulier permis de redéfinir le formalisme mis en place bien plus "proprement" que cela n'avait jamais été fait. Ne serait-ce qu'à ce titre, sa contribution est d'une importance fondamentale qu'il ne manquera sans doute pas lui-même de minimiser.

Merci aussi en général à tous les "permanents" RIA qui m'auront apporté une aide ou un soutien à des moments parfois critiques, tels que Viviane, Philippe, ou Thierry S.

Ces quatre années de thèse m'ont amené également à assurer des enseignements à l'université

[1] Robotique et Intelligence Artificielle

Paul Sabatier, donnant lieu à d'autres rencontres et collaborations d'importance. Dans ce cadre, je souhaite remercier pour leur aide et leur soutien mes collègues Emmanuelle Despontin, Gérard Mouney, Michel Combacau, Philippe Esteban, Viviane Cadenat, Cyril Briand, Patrick Danès ainsi que tous ceux avec qui j'ai été amené à travailler à l'UPS.

Merci aussi aux collègues avec qui j'ai eu la chance de travailler dans le cadre du projet R2M, en particulier l'équipe *Hylos 2* du LRP : Christophe Grand, Guillaume Besseron, Faïz Ben Amar et Frédéric Plumet.

Au cours de cette thèse j'ai été amené à travailler en collaboration avec une équipe d'un centre de la NASA, le NASA Ames Research Center, situé au coeur de la Silicon Valley en Californie, dans le cadre du projet CDS (Collaborative Decision Systems). Cette aventure a débuté en février 2005, marquée par un séjour de 2 mois et demi sur site, et conclue par des démonstrations publiques réussies les 29 et 30 septembre 2005. Ce fut une expérience très enrichissante à plusieurs points de vue, et je tiens à remercier toutes les personnes qui l'ont permise et/ou y ont participé. Merci donc à Félix, Raja et Simon pour m'avoir accordé leur confiance en me confiant cette mission ; à l'équipe IDEA, avec en particulier son "poumon" Nicola Muscettola, mon principal collaborateur Chuck Fry, ainsi que David Rijsman et Paul Tompkins, et à toute l'équipe CDS en général, dont son meneur Liam Pedersen ainsi que Vinh To et Jeffrey Iglesias. Merci aussi à Christina Stosic (MCT[2]), Sonie Lau (NASA) et l'AIPT[3] pour les (nombreuses et difficiles !) questions administratives. Thanks to all for that unique experience !

Enfin, la contribution quotidienne la plus importante vient sans nul doute des nombreux collègues doctorants, ingénieurs, stagiaires, et/ou amis avant tout, avec qui j'ai partagé tant de bons et mauvais moments, de réflexions et discussions, de remises en cause, de moments de détente et...beaucoup de café ! Les principaux se seront déjà probablement reconnus, qu'ils soient francais, mexicains, colombiens, catalan, canadien, portuguais, ou bien turc : "Chef" Aurélie, Vince, Claudia "Trufa", Arno, Nacho, Luis, Leo, Sylvain, Jérôme "Petit-Bonhomme", Efrain "Piltrafa", Ludo, Gee "El Plato", Gaviña, Maître Joan, Jean-Michel, Akin, Paulo, Maître Sylvain J., Gustavo, Martial, ainsi que mon amie Elodie, y una nena especial, une certaine Di...
Un grand merci mes amis, queridos amigos. J'espère que nous resterons longtemps en contact.

Tambien puedo añadir que estos ultimos años me sirvieron para mejorar bastante mi español (ademas del ingles gringo), y especialmente el mexicano ! Y esto gracias a los numerosos buenos momentos que pude pasar con mi estimada gente iberoamericana. Que chido/chevere fue pasarla con ustedes, amigos !

"Last but not least", merci à mes parents pour la relecture du manuscrit et surtout leur confiance et soutien inconditionnels, et tant de (bonnes) choses qu'ils m'ont apportées sans lesquelles cette thèse n'existerait pas. Je ne saurai jamais comment leur exprimer toute ma gratitude pour tout ce que je leur dois.

En résumé :
Merci beaucoup à tous !
Muchas gracias a todos !
Thanks very much to all !

[2]Mission Critical Technologies Inc.
[3]Association for International Practical Training

A KatPat,
le travail d'un petit orteil
qui voulait devenir un poil plus grand

Table des matières

Table des figures

Introduction générale

Le problème de la navigation autonome d'un robot mobile en environnements naturels a suscité un intérêt croissant ces dernières années, en particulier sous l'impulsion des missions à succès d'exploration planétaire de la NASA sur Mars, dont la dernière, Mars Exploration Rover (MER), dépasse même en ce moment les espérances initiales de ses concepteurs.

Les applications de la robotique mobile d'extérieur se multiplient. Outre l'exploration planétaire, nous pouvons citer :

- l'exploration ou l'intervention dans des milieux hostiles pour l'homme en général (zone radioactive, polaire, présence de feu...), ou simplement difficiles d'accès,
- les opérations militaires délicates telles que le déminage ou l'accompagnement de troupes,
- les missions de surveillance ou de protection.

Les roboticiens cherchent à augmenter progressivement le degré d'autonomie de leurs robots, jusqu'à arriver à une autonomie complète et fiable pour des missions de très longue durée. On peut naturellement se poser la question du besoin de cette autonomie, tant on imagine que la plupart du temps les missions nécessitant l'intervention de robots pourraient être réalisées par simple téléopération de ces derniers, et vu la grande difficulté de doter un robot d'une intelligence suffisante. D'autre part, la confiance des hommes va bien plus vers ses semblables que vers des machines, fussent-elles conçues par eux-mêmes. Ils restent en effet particulièrement méfiants de l'acquisition d'une forme d'intelligence par des entités artificielles.
Dans le cadre de l'exploration planétaire sans présence d'humains sur place, ce besoin se justifie aisément par les délais nécessaires à la communication avec la Terre. La NASA étudie également en ce moment des missions faisant intervenir à la fois des astronautes et des robots (plus gros que les mini rovers envoyés jusqu'ici) jouant le rôle d'assistants des astronautes. Même dans ce contexte, on demande au robot de disposer d'un certain degré d'autonomie, pour des missions pouvant durer quelques heures, car le temps des quelques astronautes présents sur la planète serait trop précieux pour le consacrer uniquement à la téléopération d'un robot[4].

L'augmentation de l'autonomie et de l'intelligence des véhicules atteint jusqu'à l'automobile, équipée de plus en plus de systèmes prenant des décisions de manière plus ou moins autonome en cas d'urgence, telles que corriger la trajectoire ou accentuer un freinage de sécurité, en attendant les premiers véhicules entièrement automatisés.

Le point fondamental pour un robot mobile est sa capacité à réaliser de manière effectivement autonome un déplacement d'un point à un autre (ou à une série d'objectifs), donné(s) par ses coordonnées (x, y), dans un environnement non parfaitement connu a priori. Ce problème, bien que largement étudié par les chercheurs depuis quelques décennies, ne peut encore être considéré

[4] voir le projet CDS : *Collaborative Decision Systems* du NASA Ames Research Center, auquel nous avons été amenés à collaborer au cours de cette thèse, en 2005.

comme résolu. Réaliser cette action de déplacement implique de disposer de diverses fonctionnalités (perception, décision sur le mouvement ou la trajectoire, réalisation de ce mouvement). De nombreuses méthodes différentes ont déjà été étudiées et développées pour cela. Toutefois, vu la diversité des situations qui peuvent être rencontrées en environnements extérieurs, aucune de ces méthodes ne peut prétendre permettre d'obtenir de bons résultats dans tous les cas. De plus, une grande variété de types de plates-formes ayant également été étudiées et conçues, une méthode donnée ne peut généralement pas s'adapter à toutes ces plates-formes. Par conséquent, il n'existe pas de méthode de déplacement *universelle* et idéale et il semble illusoire d'en rechercher une.

Les diverses expériences du passé (entre autres au sein du groupe RIA[5] du LAAS-CNRS) nous ont convaincu qu'un robot mobile autonome devrait disposer de plusieurs modes de déplacement, à savoir plusieurs manières de réaliser les étapes de la navigation et la locomotion, afin de pouvoir négocier le plus de situations possibles. Dès lors, conserver le caractère autonome du robot se servant de ces fonctionnalités demande qu'il soit capable de choisir en ligne de manière autonome le meilleur mode de déplacement à utiliser. Il convient bien entendu d'effectuer ce choix à partir d'une *reconnaissance de la situation* dans laquelle le robot se trouve, sachant les situations auxquelles sont dédiées chacun des modes. Il semble également judicieux de prendre en compte pour cette décision le *comportement* du robot et du mode couramment utilisé. En effet, quelle que soit la situation, il vaut mieux pour le robot trouver une autre solution plutôt qu'insister avec un mode qui ne permet pas de réaliser un déplacement satisfaisant. C'est essentiellement sur ce second aspect que se sont focalisés nos travaux.

L'objectif principal de cette thèse est donc de développer un formalisme de sélection en ligne du *mode de déplacement à utiliser*. Dans ce cadre, les contributions de cette thèse sont :

- le développement d'un formalisme probabiliste de sélection de mode de déplacement (navigation/locomotion) pour un robot mobile autonome évoluant en environnements naturels, sur la base d'information de *contexte* et de *comportement* ;
- l'adaptation ou le développement de modes de navigation et de locomotion pour un tel robot, ainsi que leur intégration ;
- la mise au point d'une représentation du terrain en terme de *contexte* lié aux modes de navigation disponibles (reconnaissance de la situation dans laquelle le robot se trouve) ;
- le développement de *moniteurs* pouvant évaluer le comportement de chaque mode et fournir, après combinaison, des informations utiles au système de sélection de modes ;
- enfin, des tests intégrés effectuant des changements de mode en ligne.

Le plan de ce mémoire suit globalement ce schéma.

Le **chapitre 1** vise à développer la problématique et le contexte et à positionner notre approche dans l'état de l'art. Après une introduction à la notion de navigation autonome en extérieur pour un robot mobile, son principe et ses difficultés, il évoque différents exemples de méthodes de locomotion et de réalisations de fonctionnalités de navigation proposées dans la littérature, avant de présenter brièvement des systèmes robotiques multi-modes développés récemment. Enfin, les raisons des bases de notre approche sont avancées suite à cet état de l'art.

Le **chapitre 2** présente le formalisme de la sélection de mode de déplacement. Après un rappel des notions de probabilités, systèmes stochastiques et de diagnostic que nous exploitons, puis un positionnement bibliographique, nous proposons notre approche probabiliste pour la sélection

[5] Robotique et Intelligence Artificielle

en ligne du mode à appliquer, sur la base des données de contexte et de comportement.

Le **chapitre 3** évoque les modes de déplacement exploités dans ces travaux. Il présente tout d'abord les plates-formes robotiques ayant constitué les supports d'expérimentations puis le fonctionnement des modes de navigation et de locomotion intégrés. Certains ont dus être adaptés à nos plates-formes et à nos besoins, d'autres développés. Les objectifs de surveillance du comportement de ces modes nécessitent que ces fonctionnements soient évoqués avec suffisamment de détails. Des systèmes multi-modes construits sur le formalisme précédent et reposant sur ces modes sont enfin proposés.

Le **chapitre 4** comporte une description de la représentation du terrain que nous avons développée pour fournir au système de sélection de modes les données de contexte. Elle est précédée d'une évocation des représentations approchantes.

Le **chapitre 5** est consacré aux *moniteurs* chargés de la surveillance du comportement des modes de navigation et de locomotion, et fournissant donc les données de comportement au système de sélection de modes.

Le **chapitre 6** présente les expérimentations réalisées et résultats obtenus avec le système complet intégré. Les systèmes multi-modes proposés dans le chapitre 3 sont tout d'abord complétés en intégrant les moniteurs. Les résultats sont alors discutés, tout comme le formalisme de notre approche.

Enfin, une conclusion générale propose un bilan de ces travaux de recherche ainsi que des extensions et perspectives envisagées.

Quelques détails supplémentaires sur certaines parties figurent dans les annexes à la fin de ce document.

Chapitre 1

Contexte et positionnement

Ce chapitre présente le contexte et les raisons de notre travail, tout en le positionnant dans le spectre actuel de la robotique mobile. Après une introduction à la robotique mobile en environnements naturels, nous présentons quelques exemples de plates-formes disposant de capacités de locomotion diverses, puis des fonctionnalités de navigation, et enfin nous introduisons plus avant les motivations de ce travail sur la base d'exemples illustratifs, ainsi que les principales composantes de son contenu.

1.1 Navigation autonome d'un robot mobile en environnement naturel

La navigation d'un robot mobile autonome en environnements naturels est un problème difficile et encore loin d'être résolu, en particulier du fait de la diversité des situations que peut rencontrer un rover[6] et de l'absence, dans le cas général, de connaissance précise *a priori* sur les terrains rencontrés. Il convient ici de rappeler les principes de résolution de ce problème, de mentionner quelques applications et d'insister sur les difficultés qui demeurent.

1.1.1 Principe général

Réaliser une tâche telle que *se déplacer vers le point de coordonnées (x,y)*, aussi simple puisse-t-elle paraître à un humain, requiert la mise en oeuvre de fonctionnalités potentiellement complexes de *perception/décision/action* (figure 1.1). Le rôle de ces fonctionnalités est le suivant :

- **perception :** Il s'agit essentiellement de détecter les obstacles et d'effectuer éventuellement une modélisation de l'environnement (3D dans notre cadre d'environnements naturels) pour fournir et mettre en forme les informations nécessaires à la *décision* sur le déplacement à réaliser.
- **décision :** L'environnement étant perçu, et éventuellement modélisé, il faut décider le type de mouvement à effectuer en générant des consignes de vitesses appropriées à envoyer au robot (pour un mouvement réactif par exemple) ou en choisissant une trajectoire à exécuter.
- **action :** Il s'agit alors de veiller à réaliser le mouvement ou la trajectoire décidé, suivre les consignes de vitesses reçues en appliquant par exemple le type de commande adapté.

[6] Un *rover* peut être défini comme un véhicule autonome destiné à l'exploration de la surface d'un corps extra-terrestre (comme la Lune ou Mars). Nous utiliserons ce terme dans ce manuscrit plus généralement pour désigner un véhicule autonome à roues évoluant dans un environnement naturel, qui peut également être terrestre.

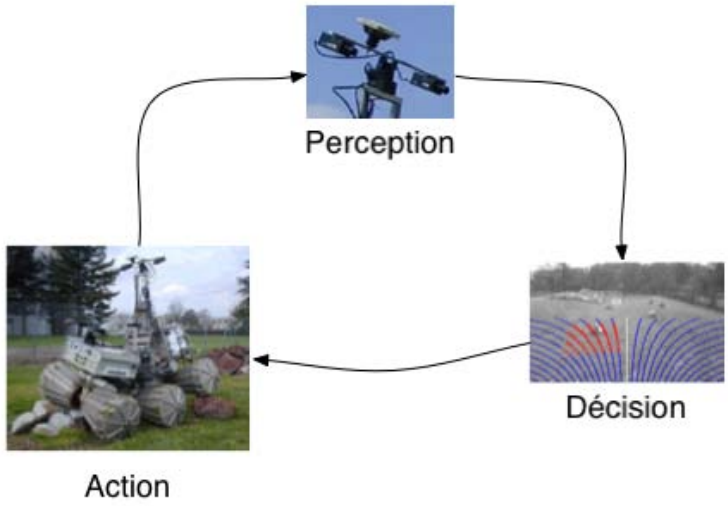


FIG. 1.1 – Le cycle perception/décision/action que doit réaliser un robot autonome pour réaliser une navigation.

1.1.2 Applications

Les applications de cette discipline se multiplient, au-delà de l'exploration planétaire et martienne en particulier. Elles concernent essentiellement des zones pratiquement inaccessibles pour l'homme et/ou hostiles. Des projets récents ont ainsi déployé de tels robots en Antarctique ou dans des déserts très arides, ou encore pour des missions de déminage [Coronado-Vergara et al., 2005]. Les développements de ces dernières années ont permis de mettre en oeuvre des projets de plus en plus audacieux, avec des rovers disposant de capacités d'autonomie accrues. Deux exemples notables de projets récents de robotique mobile autonome en environnements naturels sont :

- Le programme *Mars Exploration Rover* (MER[7]) [JPL-NASA, 2006], mené par le Jet Propulsion Laboratory (JPL) de la NASA et ses deux rovers Spirit et Opportunity chargés d'explorer une partie de la surface martienne à la recherche de traces d'eau, et donc de vie.
- Le projet *Life in the Atacama* [Wettergreen et al., 2005][CMU-NASA, 2006] mené par l'université américaine de Carnegie Mellon (CMU) en collaboration avec le Nasa Ames Research Center. Les rovers de ce projet, Fido en 2003 (figure 1.2), puis son successeur Zoe en 2004-2005, ont réalisé des missions de navigation dans le désert de l'Atacama au Chili pour trouver et analyser des éléments vivants dans cette zone la plus aride et parmi les plus hostiles sur Terre.

Tout aussi réussis soient-ils, ces projets démontrent néanmoins qu'il reste beaucoup de chemin à parcourir avant d'arriver à la réalisation de plates-formes disposant d'une autonomie complète et suffisamment fiables pour des missions à long terme. En effet, concernant par exemple les deux rovers du programme MER, d'une part la plupart des mouvements sont décidés au sol et d'autre part les mouvements autonomes se ramènent à de simples évitements d'obstacles sur de courtes distances.

[7] *Mars Exploration Rover* Mission, NASA : Programme qui a connu son "envol" les 10 juin et 7 juillet 2003 avec le lancement des sondes contenant les deux rovers jumeaux Spirit et Opportunity, puis leurs atterrissages respectifs sur le sol martien les 4 et 25 janvier 2004.

1.1.3 Difficultés rencontrées

Les difficultés sont nombreuses. Outre celles inhérentes à l'autonomie d'un robot mobile en général, celles liées au contexte d'environnements naturels (par opposition aux environnements structurés) sont principalement le manque d'information *a priori* sur les terrains traversés (les informations initiales, si disponibles, sont généralement très imprécises) et l'impossibilité d'effectuer des hypothèses sur une structure de l'environnement (présence de murs, de portes etc...). Les auteurs de [Kelly et al., 1997] précisent ainsi que les environnements extérieurs et intérieurs diffèrent sensiblement dans la stratégie de détection d'obstacles qu'il convient d'adopter : en intérieur, en l'absence d'une perception signifiant le contraire on peut considérer l'espace comme libre d'obstacles, alors qu'en extérieur l'absence d'information amènera à considérer par défaut la zone comme non franchissable (obstacle), et ce jusqu'à preuve du contraire. Une autre difficulté majeure réside dans la grande variété des situations que le robot peut rencontrer : on peut rencontrer des terrains plus ou moins accidentés, et de natures aussi différentes que des dunes de sables, un sol enneigé ou même glacé, boueux ou constitué de terre sèche etc...

(a) Dunes de sable (b) Terrain accidenté

(c) Fido dans le désert de l'Atacama (Chili) (d) Nomad en Antarctique

FIG. 1.2 – Exemples d'environnements naturels variés

La diversité des situations possibles demande la mise en oeuvre de fonctionnalités adaptées, et le besoin de multiplier ces possibilités. En effet, nous pensons que l'on ne peut raisonnablement pas espérer obtenir des résultats pleinement satisfaisants dans toutes les situations avec une seule réalisation des capacités de *perception/décision/action*. Pour garantir l'autonomie du rover il faut donc un système permettant au robot de choisir la fonctionnalité la mieux adaptée à la situation actuelle.
Pour des questions de lisibilité, nous séparons ici ces fonctionnalités en deux catégories : les

fonctionnalités dites de **navigation** (concernant essentiellement la *perception* et la *décision*) et celles dites de **locomotion** (*action* : exécution du mouvement). Ainsi, nous présentons tout d'abord des systèmes de locomotion présentant des caractéristiques et utilités différentes avec des exemples de plates-formes, puis nous verrons quelques exemples de fonctionnalités de navigation en extérieur.

1.2 Systèmes de locomotion

L'un des éléments essentiels pour garantir une bonne performance d'un robot autonome en extérieur est la structure de *locomotion* de la plate-forme utilisée.

Les systèmes de locomotion terrestre robotisés peuvent être de trois différents types principaux [Grand, 2004] : à roues ou chenilles, à pattes (bien souvent sur le modèle d'animaux ou de l'humain) ou encore apodes (utilisation d'éléments de la structure du robot pour générer et appliquer au sol les efforts de propulsion). Nous nous intéresserons plutôt dans cette étude aux systèmes à roues, plus proches de nos applications, voire à quelques systèmes hybrides, même si les autres types de locomotion présentent également un intérêt manifeste et croissant. Citons par exemple le robot M-TRAN (Modular Transformer) [Kamimura et al., 2002, Kurokawa et al., 2002] développé par l'AIST[8], du Tokyo Institute of Technology. Cet engin présente des capacités de reconfiguration complète lui permettant de se déplacer dans des modes de locomotions très différents. Il est composé de modules constitués chacun de deux demi-cylindres connectés par un lien magnétique. Grâce à ces modules le M-TRAN peut par exemple passer d'une forme de robot quadrupède (progressant en marchant) à celle d'une chenille (se déplaçant donc en rampant, voir figure 1.3).

FIG. 1.3 – Le robot reconfigurable MTRAN-II de l'AIST, université de Tokyo

Concernant les systèmes de robotique mobile à roues, la variété dans le type de locomotion peut venir d'un des éléments suivants :

- le système de propulsion
- le système de direction
- le système de suspension

1.2.1 Le système de propulsion

Concernant les robots à roues, la propulsion la plus répandue est bien entendu celle obtenue par simple rotation des roues. Elle peut également provenir de l'actionnement d'articulations internes du châssis, si la plate-forme en possède (voir par exemple le robot Lama lorsqu'il se

[8]National Institute of Advanced Industrial Science and Technology

déplace en mode *péristaltique*[9], section 3.3.2).
Certains projets ont imaginé des systèmes plus originaux comme le Cliff-bot [Pirjanian et al., 2002], développé par le JPL, qui est doté d'une possibilité de traction par un système auxiliaire de grappin-treuil.

1.2.2 Le système de direction

Des exemples de systèmes classiques de direction pour les robots à roues sont :

- *la rotation par vitesses différentielles* (*skid-steering*) : ce type de direction est souvent utilisé sur les plates-formes ne disposant pas de roues directrices (axes de rotation des roues non orientables). Il consiste à imposer un différentiel de vitesse sur les roues (ou chenilles dans le cas d'un char) situées de chaque côté du châssis afin d'obtenir la rotation de celui-ci. Le principal problème de ce système est la génération de forts glissements, augmentant ainsi sensiblement la consommation énergétique de la plate-forme et introduisant de fortes erreurs supplémentaires dans la localisation par odométrie, déjà relativement peu fiable dans des terrains extérieurs, surtout lorsqu'ils sont accidentés.
- *la rotation à châssis articulé* : dans ce cas, le châssis lui-même contient une ou des parties orientables.
- *la rotation avec roues directrices indépendantes*. De tels systèmes offrent de multiples possibilités de rotation et permettent des déplacements singuliers comme le déplacement "en crabe". Néanmoins ils augmentent la complexité mécanique et rendent souvent les structures moins robustes que des systèmes à rotation par vitesses différentielles par exemple.

Un exemple notable de système de direction original est le robot Nomad de CMU/NASA (voir figure 1.4).

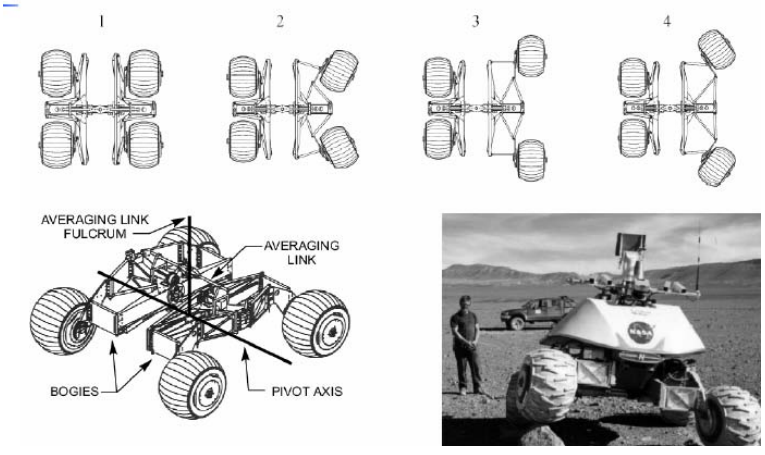


FIG. 1.4 – Le robot Nomad (CMU/NASA) et son système de direction

Certaines plates-formes disposent de systèmes de direction hybrides, permettant ainsi de choisir le système le mieux adapté à la situation courante, voire d'en combiner plusieurs pour augmenter les possibilités de négocier une difficulté.

[9] dans ce mode le robot "rampe" comme une chenille.

1.2.3 Le système de suspension

Le système de suspension sur une plate-forme de locomotion tout-terrain permet d'augmenter ses capacités de franchissement (et sa stabilité) en garantissant le maintien du contact des roues avec le sol pour garder la meilleure traction possible. Ces suspensions peuvent être passives ou actives. Dans ce dernier cas, il est possible de développer diverses lois de commande tirant parti de ces suspensions actives pour améliorer les performances de locomotion du système (en optimisant des critères tels que la posture de la plate-forme, par exemple) [Grand, 2004].

1.2.4 Exemples de robots mobiles à roues pour la locomotion en extérieur

Ces dernières années, le développement de plates-formes robotiques pour le déplacement en extérieur s'est multiplié, en particulier dans le but d'obtenir des systèmes aux capacités de franchissement accrues. Nous présentons dans cette partie quelques exemples qui nous paraissent intéressants, qui sont principalement de deux types : les systèmes à mobilités internes passives et ceux, au contraire, à mobilités internes actives. Concernant la locomotion, ce sont ces derniers qui nous intéresseront plus particulièrement dans le cadre de ces travaux, de part leurs possibilités multimodales.

Systèmes à mobilités internes passives

Ces systèmes sont destinés à épouser au mieux la forme du terrain, afin de conserver un bon contact des roues avec le sol, et donc de bonnes capacités de traction. Parmi ces robots, citons les rovers martiens Rocky 8 et ses successeurs Spirit et Opportunity du programme MER ou encore le robot Shrimp développé à l'Ecole Polytechnique Fédérale de Lausanne (EPFL) [Estier et al., 2000], système à 6 roues capable de franchir des marches d'une hauteur égale à deux fois le diamètre de ses roues.

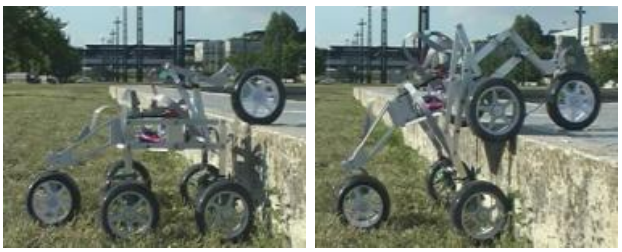


Fig. 1.5 – Le robot Shrimp (EPFL)

Systèmes à mobilités internes actives

Ces systèmes sont munis de châssis comportant des liaisons internes qui peuvent être actionnées. Cela permet la réalisation de différentes lois de commande pour réaliser des tâches de franchissement plus ou moins spécifiques, ou d'optimiser la configuration du châssis afin d'améliorer le comportement de la locomotion sur terrain accidenté ou encore de minimiser la consommation d'énergie. Certains robots disposent par exemple d'essieux articulés les uns par rapport aux autres, leur permettant de se déplacer dans un mode tel que le *péristaltisme* [Andrade et al., 1998]. C'est le cas des châssis de type Marsokhod, mis au point par les russes de VNII Transmach, comme le robot Lama du LAAS dont nous reparlerons plus loin (section

3.1.2). Parmi ces robots (fig. 1.6) nous trouvons des plates-formes hybrides roues-pattes (les roues se trouvent à l'extrémité de parties mécaniques de type pattes) telles que les robots SRR (JPL/CalTech), WorkPartner (HUT[10]) ou Hylos du Laboratoire de Robotique de Paris (LRP) [Grand, 2004][Grand et al., 2002].

(a) SRR (JPL/NASA) (b) WorPartner (HUT) (c) Hylos (LRP)

FIG. 1.6 – Plates-formes à mobilités internes actives

Certains robots combinent des mécanismes locomoteurs bien différents, tels que le Roller-Walker de l'Université Technologique de Tokyo ou Azimut de l'Université de Sherbrooke, Canada, qui disposent de véritables capacités de locomotion multimodale.

Le Roller-Walker [Endo et al., 1999][Endo et al., 2000] (fig. 1.7) est équipé de quatre pattes comportant chacune une roue libre. Ces roues peuvent être en position verticale, permettant au robot de rouler, ou bien en position horizontale, se transformant alors en pieds. Ce robot dispose ainsi de deux modes de locomotion bien distincts :

- le mode dit *patinage* (skating) dans lequel le robot roule, propulsé par l'action de ses pattes (les roues ne disposent pas d'actionneurs propres). Ce mode est plutôt adapté au déplacement en terrain assez plat et régulier.
- le mode de *marche* (walking), qui convient mieux aux déplacements sur terrains accidentés.

Azimut [Michaud et al., 2003] dispose de quatre éléments de propulsion comportant deux articulations et une courroie de traction, constituant chacun un système hybride roue-patte-chenille (cf. figure 1.8). Trois modes sont possibles :

- le mode roulement de type *char à chenilles* lorsque les roues sont parallèles au sol,
- le mode *roulement sur roues* lorsque les pattes sont verticales orientées vers le haut,
- le mode *marche sur pattes* lorsque celles-ci sont orientées vers le bas (courroie de traction à l'arrêt).

Le robot WorkPartner (HUT) est un robot hybride roue-patte disposant de deux modes de locomotion : le roulement simple et le mode dit *rolking*, qui combine mouvement des pattes et rotation des roues pour la traction. Ce mode est mieux adapté à des terrains peu cohésifs tels que le sable ou la neige.

Nous avons vu quelques exemples de différents types de plates-formes de robotique mobile dédiées à la navigation en extérieur, en particulier sur terrain accidenté. Certaines présentent en particulier des modes de déplacement distincts afin de mieux faire face à la variété de situations

[10] Helsinki University of Technology, Automation Technology Laboratory

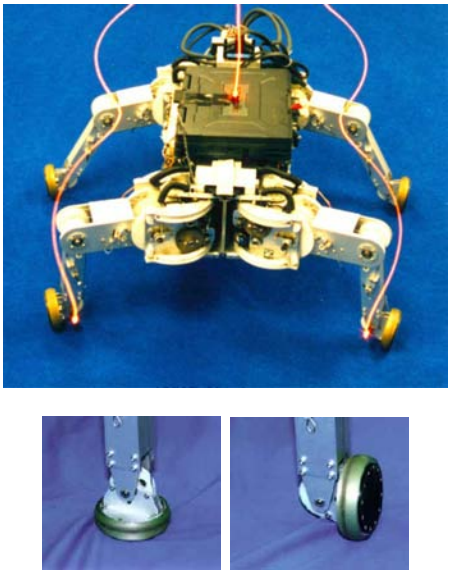


FIG. 1.7 – Le Roller-Walker de l'AIST

FIG. 1.8 – Azimut (université de Sherbrooke)

qu'elles sont susceptibles de rencontrer. Nous voyons également apparaître le fait que ces modes sont en général plutôt dédiés à certains types de contexte.

1.3 Fonctionnalités de navigation

Le problème (ainsi que les avantages) de la multiplicité des fonctionnalités ne s'arrête pas à la locomotion. Il concerne aussi les méthodes de *perception* ou de choix de mouvement et de trajectoire (*décision*). Nous présentons dans cette partie quelques exemples de fonctionnalités de navigation : tout d'abord des méthodes de perception et de représentation de l'environnement puis des techniques de génération de déplacement.

1.3.1 Perception et représentation de l'environnement

Les fonctions de perception sont bien entendu essentielles à la réalisation de machines intelligentes : c'est en effet par leur biais que ces dernières peuvent appréhender l'état du monde, et ainsi adapter la résolution et l'exécution des tâches utiles pour réaliser un déplacement.

Capteurs

En environnement extérieur, les capteurs les plus couramment utilisés pour la perception de l'environnement sont les suivants.

Les capteurs lasers fournissent une mesure proximétrique des obstacles avec une bonne précision sur la distance à l'objet mesurée. Cependant, les lasers 2D, n'effectuant qu'un balayage dans un plan, ne sont exploitables que sur terrain plat et à condition que les obstacles soient visibles à la hauteur du laser. Ils sont ainsi très performants et couramment utilisés dans les environnements structurés comportant des obstacles tels que des murs, voire des environnements

semi-structurés[11], mais moins courants dans les applications d'extérieur. Les lasers 3D, fournissant une représentation de l'environnement plus complète et plus adaptée à notre contexte, sont toutefois encore peu utilisés en pratique à cause de leur coût encore élevé et de leur lenteur.

Les caméras sont sans doute les capteurs capables de fournir l'information la plus dense sur l'environnement, d'autant plus depuis l'exploitation de la couleur. Lorsque l'on utilise deux caméras montées en un banc de *stéréo-vision* on peut estimer la profondeur de chaque pixel *apparié* dans les paires d'images et ainsi associer à chaque point des coordonnées 3D complètes [Lemonde, 2005]. Une acquisition d'images par un banc de stéréo-vision permet donc de générer un nuage de points 3D perçus dans l'environnement, qui peut être exploité pour construire un *Modèle Numérique de Terrain* ou *carte d'élévation*. Les temps de calcul nécessaires avant d'arriver à cette étape obligent néanmoins le plus souvent à des déplacements relativement lents, même si des travaux récents de réalisation de capteurs de stéréo-vision avec algorithmes associés intégrés sur carte FPGA [Boizard et al., 2005] voire l'apparition de nouveaux types de capteurs (capteurs à temps de vol) tendent à réduire sensiblement ce problème. Notons enfin l'intérêt croissant pour les caméras panoramiques, offrant une vue plus complète de l'environnement.

Les capteurs infra-rouge sont en général utilisés en complément d'autres capteurs tels que des caméras "traditionnelles", par exemple pour faciliter l'extraction des obstacles dits *négatifs*[12] [Matthies et al., 2003].

Représentation de l'environnement

Différents types de représentation de l'environnement peuvent être utilisés [Rives et al., 2001a]. La plupart de ces représentations sont exploitées pour effectuer de la détection et de l'évitement d'obstacles, voire de la localisation, particulièrement dans le cas du SLAM (Simultaneous Localization And Mapping[13]). Nous présentons quelques exemples de méthodes de représentation, en nous focalisant sur celles utilisées en environnement naturel.

Carte d'occupation : Une grille d'occupation est une représentation discrète 2D d'un espace navigable. L'environnement est ainsi discrétisé en cellules auxquelles est attribué un vecteur décrivant son état. La principale composante de ce vecteur est généralement la probabilité d'occupation de la cellule, permettant d'inférer la présence ou non d'un obstacle. Cette grandeur est obtenue à partir des données de perception, typiquement de télémétrie, fusionnées au cours des acquisitions successives réalisées dans le temps. Ce type de représentation est plutôt dédié aux espaces navigables supposés plans.

Carte d'élévation : Ce type de représentation est bien adapté aux besoins de la navigation en terrain accidenté. Une carte d'élévation (nommée aussi Modèle Numérique de Terrain, et souvent désignée par le sigle MNT) est fondée sur une discrétisation du plan horizontal en cellules régulières. A chaque point discrétisé (x, y) du sol est associée une altitude z. Ainsi les MNT sont souvent qualifiés de *carte 2,5 D* plutôt que 3D. Ce modèle est généralement obtenu

[11] On appelle généralement *environnements semi-structurés* des environnements d'extérieur initialement naturels qui ont été modifiés par l'homme pour tracer des voies de circulation (chemins, routes), délimiter des zones (par des marques ou clôtures) etc...

[12] Les obstacles négatifs sont définis comme des trous, fossés ou autres dépressions dans le sol.

[13] Localisation et cartographie simultanées : méthodes consistant à réaliser les deux tâches de localisation et de cartographie en parallèle, chaque tâche exploitant les informations générées par l'autre.

par fusions successives des acquisitions réalisées par des systèmes capables de générer des points 3D : télémètre laser 3D ou banc de stéréo-vision.

Modèle qualitatif de l'espace libre : Cette représentation, également de type carte discrète de l'environnement, propose des cartes *étiquetées*, permettant le plus souvent de statuer sur le caractère navigable ou non de chaque cellule de la carte, en effectuant une classification sur la base d'attributs géométriques, possiblement complétés par des informations complémentaires telles que des attributs de texture. Une procédure de regroupement des pixels peut ensuite permettre d'obtenir un graphe de régions exploitable pour la navigation.

Représentation de la topologie de l'environnement : Bien que plus facilement exploitable en environnement structuré d'intérieur (car une topologie y est plus facilement définissable et identifiable) des utilisations de ces méthodes ont également été étudiées pour des environnements naturels. Des travaux proposent par exemple une carte cognitive décomposée en *places* particulières dont chacune contient des amers, ou encore des approches de représentation d'un terrain supposé plat par morceaux constituée de zones planes et d'amers.

Méthodes de localisation

La localisation est une des fonctions fondamentales de la robotique mobile. En effet il est indispensable de disposer d'une estimée de la position du robot à chaque instant afin que ce dernier puisse réaliser une trajectoire sans collision, suivre fidèlement les consignes de déplacement reçues et bien entendu atteindre le point objectif. Devant l'importance du problème, de nombreuses méthodes de localisation ont été développées ces dernières années. Certaines ne sont applicables que dans un type de contexte particulier et doivent donc être choisies judicieusement. D'autres au contraire en sont indépendantes. Si plusieurs méthodes sont disponibles, il convient de les combiner en effectuant si possible une fusion de leurs estimations, ce qui permet d'améliorer la qualité de l'estimée finale. Nous présentons brièvement ici quelques-unes des méthodes utilisées en environnement naturel [Rives et al., 2001b] :

Odométrie : L'odométrie consiste à estimer des positions relatives à partir de la mesure du déplacement angulaire des roues fournie par des codeurs optiques ([Mallet, 2001], chapitre 2). Ces codeurs sont très répandus chez les robots à roues, spécialement les robots de type "char" (effectuant des rotations par vitesses différentielles, cf. section 1.2.2), et la mise en oeuvre de cette méthode est relativement simple, la rendant très courante. Cependant, elle présente une forte dérive de l'erreur, problème affectant toutes les méthodes de localisation dites *à l'estime*[14] en général. Les causes principales en sont les deux types d'erreurs rencontrées et la difficulté de les modéliser :

- les erreurs *systématiques* [Borenstein et al., 1996b] sont intégrées à chaque itération sous la forme d'une erreur additive. Elles sont dues principalement aux biais sur la valeur de paramètres du robot tels que les rayons des roues ou la longueur de l'entraxe, mais aussi aux erreurs de mesure (résolution du codeur, fréquence d'échantillonnage).
- les erreurs *occasionnelles* sont essentiellement liées à la validité de l'hypothèse de roulement sans glissement sur un sol localement plan, sur laquelle reposent les équations de l'odométrie classique [Peynot, 2002]. Les nombreux glissements qui se produisent en réalité, particulièrement dans les environnements naturels et sur terrain accidenté, voire peu

[14] *dead reckoning* en anglais

cohésif, provoquent ainsi d'importantes erreurs occasionnelles qui sont pratiquement impossibles à évaluer et corriger sans utiliser une méthode de localisation externe et absolue. Certaines études ont tout de même permis de mettre en évidence quelques compensations, néanmoins insuffisantes (voir par exemple [Peynot et al., 2003]). C'est ce type d'erreurs qui prédomine largement dans notre contexte d'environnements naturels.

Afin de réaliser une localisation 3D sur la base d'informations odométriques, il est nécessaire d'ajouter des mesures d'attitudes fournies par un des inclinomètres ou par intégration de mesures inertielles. De plus, concernant les robots mobiles de type "char", une part importante des erreurs dites *occasionnelles* provenant de l'estimation erronée de la rotation du robot (d'importants glissements se produisent, par essence, lors de la rotation d'une telle plate-forme), on peut avantageusement exploiter une estimation de cette rotation par intégration de la mesure d'un gyromètre [Mallet, 2001], réalisant ainsi une méthode "hybride" parfois nommée *gyrodométrie* [Borenstein et al., 1996a].

Localisation par capteurs inertiels : Cette méthode exploite les informations fournies par une *centrale inertielle*, appareil composé de trois gyromètres mesurant chacun la vitesse de rotation autour d'un axe et de trois accéléromètres mesurant chacun une accélération suivant un axe. Il est ainsi possible de reconstituer une estimation de la position 3D du véhicule qui embarque cet appareil après intégration des données des gyromètres et double intégration des accélérations. Cela rend toutefois cette estimation très sensible aux bruits de mesures et soumise à une dérive importante si les informations ne sont pas combinées à d'autres données. Ainsi les chercheurs travaillant sur les problèmes de localisation utilisent-ils bien souvent un système *hybride* fusionnant les données issues du GPS et de la centrale inertielle [Sukkarieh, 2000].

Estimation visuelle de mouvement (odométrie optique) : Les données de base exploitées par cette méthode sont les couples d'images fournies par un banc de stéréo-vision. A partir d'une technique d'appariement de pixels présents dans les deux images (d'où l'on peut déduire des points 3D) puis d'un suivi entre ces pixels dans deux images consécutives au cours du déplacement, on peut estimer le mouvement apparent dans l'image et ainsi retrouver une estimée des six paramètres 3D du déplacement réel effectué [Mallet, 2001]. Cette méthode présente elle aussi une dérive due à l'intégration de déplacements élémentaires (d'où son appellation d'*odométrie optique*). Son efficacité dépend essentiellement de la sélection des *points d'intérêt* qui seront utilisés pour réaliser les *appariements* amenant à l'estimation du mouvement. [Lacroix et al., 2004] ont proposé récemment une version améliorée de cette méthode, s'affranchissant notamment de la prédiction de mouvement nécessaire auparavant et réduisant sensiblement le nombre et la pertinence des pixels utilisés pour la corrélation, permettant ainsi d'accélérer la fréquence de traitement et de production de l'estimée de déplacement.

Localisation sur amers : Les méthodes utilisant des amers ont l'avantage d'être des méthodes *absolues*, à erreurs bornées indépendantes du temps ou de la distance parcourue (dans le cas où les amers sont cartographiés au préalable[15]). Les amers[16] sont des objets, formes ou points notables dans l'environnement, dans le sens où l'on peut les reconnaître et les repérer. Ils peuvent être par exemple des balises qui ont été placées initialement en des positions connues, ou des amers "naturels" tels que des poteaux, arbres ou rochers singuliers. La difficulté de ces

[15] ce n'est pas le cas pour du SLAM

[16] Un *amer*, terme issu du vocabulaire maritime, est défini comme un "objet, bâtiment fixe et visible situé sur une côte et servant de point de repère pour la navigation", d'où le mot plus "explicite" de *landmark* en anglais.

méthodes réside dans la recherche et le suivi de ces amers (au moins quelques-uns doivent rester visibles en permanence lors du déplacement) et la mise en correspondance des amers perçus avec les amers connus.
La localisation par un système tel que le GPS[17] et ses équivalents peut être classée dans cette catégorie, en considérant la constellation de satellites comme des amers de l'environnement. Ce système est mondialement connu et une grande précision (quelques centimètres) peut désormais être atteinte dans un contexte civil grâce à des systèmes de GPS différentiels, à condition d'installer une base de référence. Néanmoins, un tel système reste relativement coûteux et ce moyen de localisation n'est pas toujours disponible : c'est une évidence dans un contexte tel que l'exploration planétaire mais il peut y avoir également des difficultés sur Terre à cause de la présence d'éléments obstruants (murs, arbres...) empêchant d'être en vue d'un nombre suffisant de satellites pour obtenir une localisation fiable et précise. D'autre part, un système GPS ne fournit qu'une estimation de position 3D partielle (uniquement le triplet (x, y, z)), il faut donc le combiner avec un système capable de fournir une estimée de l'attitude et du cap du robot. On utilise le plus souvent pour cela une centrale inertielle et un compas magnétique.

1.3.2 Génération des déplacements

Une fois que le robot dispose d'une représentation de l'environnement et d'une estimée de sa position, il convient de s'en servir pour générer les déplacements qui lui permettront d'atteindre son objectif. Pour cela également, plusieurs méthodes sont possibles.

Mouvement planifié : choix d'une trajectoire

Planifier une trajectoire représentant le déplacement complet sans collisions à réaliser par la plate-forme mobile pour aller de la position initiale à la position but est un problème déjà largement traité dans la littérature de robotique mobile [Laumond, 2001]. Cependant, réaliser une telle opération suppose une bonne connaissance de la géométrie de tout l'environnement, limitant ainsi les applications aux environnements d'intérieur structurés voire d'extérieur semi-structurés (de type parking ou route par exemple). En environnement naturel, la méconnaissance de l'environnement à des distances relativement grandes du robot amène plutôt à effectuer une planification dans une structure de type graphe de régions (1.3.1) générant une série de *points de passages* qui sont des objectifs locaux transmis à une méthode de planification locale ou de mouvement réactif.

Planification locale

A partir d'une connaissance de l'environnement relativement proche du robot, sous la forme d'un MNT ou encore d'une carte classifiée, on peut réaliser une planification locale du mouvement en choisissant une trajectoire simple telle qu'un arc de cercle [Bonnafous et al., 2001]. Si cette planification simple peut être renouvelée périodiquement à une fréquence relativement élevée, le principe se rapproche d'une méthode *réactive*. Nous verrons avec plus de détails une méthode de ce type exploitée dans nos travaux dans la section 3.2.2.

[17] Global Positioning System : Système de positionnement sur le globe terrestre par triangulation des signaux émis par des satellites géo-stationnaires.

Mouvements réactifs

Ces méthodes n'effectuent au contraire aucune planification, ne faisant aucune *prévision* sur les situations à venir, dans l'espace et dans le temps. Les déplacements sont générés uniquement sur la base de règles de réaction à la présence d'obstacles détectés, dans le but premier de les éviter. Citons les méthodes classiques de champ de vecteur [Borenstein et al., 1991] ou du puits de potentiel. Nous verrons également une méthode réactive exploitée dans nos travaux (section 3.2.1).

Asservissement visuel

Il s'agit dans ce domaine de générer des mouvements permettant de suivre une cible remarquable présente dans la scène. Contrairement aux autres méthodes, les déplacements sont ici référencés par rapport à ces objets perçus, plutôt qu'à une carte de l'environnement [Espiau et al., 1992].

1.4 Exemples de systèmes multi-modes

Nous avons vu dans les sections précédentes que de nombreuses fonctionnalités permettant le déplacement autonome d'un robot mobile (représentant diverses modalités de perception, génération de mouvement et locomotion) ont été développées ces dernières années, permettant ainsi d'aborder une plus grande variété de situations. Nous sommes convaincus que pour être doté d'une réelle autonomie destinée à des missions à relativement long terme un robot doit disposer de plusieurs fonctionnalités complémentaires de *perception/décision/action*, et donc également des capacités de choix des mieux adaptées à la situation courante. Toutefois, il existe à notre connaissance encore très peu de systèmes robotiques capables de choisir de manière autonome entre ces différentes manières de se déplacer, et ces derniers sont presque toujours destinés à se déplacer dans des environnements structurés d'intérieur. Nous verrons dans cette partie comment un tel système peut être mis en oeuvre et sur la base de quelles données il convient d'effectuer cette sélection de modes.

1.4.1 Combinaison de modalités sensori-motrices par planificateur

Un exemple d'un tel système est proposé par Benoit Morisset dans sa thèse de doctorat [Morisset, 2002]. Il présente un robot, Diligent (fig. 1.9), doté de *modalités* sensori-motrices complémentaires et d'un automate de supervision sélectionnant en ligne la modalité à utiliser. Une *modalité* concerne l'éventuelle planification de chemins, la méthode d'exécution du mouvement et la méthode de localisation.

Les cinq modalités considérées dans ce travail sont différentes combinaisons de méthodes de mouvement (planification de chemins, de points de passage ou mouvement réactif) et de localisation (sur segments avec carte initiale, par odométrie recalée par une localisation sur amers visuels).

L'automate de supervision des modalités, formellement un Procédé Décisionnel de Markov [White, 1993], est construit sur la base d'un état de supervision abstrait et de transitions non déterministes étiquetées par la modalité exécutée pour cette transition. L'état de supervision est constitué de variables traduisant : le niveau d'encombrement et la complexité géométrique du lieu, l'imprécision et la confiance associées à la localisation du robot, la "couleur" topologique

FIG. 1.9 – Le robot Diligent (LAAS-CNRS) et ses modalités sensori-motrices

du lieu courant[18], la modalité en cours d'exécution, la distance parcourue depuis le départ de la navigation, ou encore l'appartenance ou non à une zone de démarrage d'une modalité.

Cet automate est généré par un apprentissage non supervisé réalisé tout au long des navigations du robot au cours desquelles des paramètres (probabilité d'occurrence et coût des transitions) sont appris, permettant d'obtenir une caractérisation des modalités utilisées, du moins *dans le contexte courant*. En effet, tout changement de lieu nécessite de reprendre la phase d'apprentissage avant de pouvoir obtenir un automate opérationnel. Bien entendu la réussite d'un tel système repose en bonne partie sur le fait que l'environnement est *structuré*, un tel apprentissage nous paraissant peu réalisable dans des environnements d'extérieur, par nature non structurés. Les modalités elles-mêmes exploitent cette caractéristique de l'environnement, ainsi que l'état de supervision, par l'intermédiaire de caractéristiques telles que la topologie du lieu.

1.4.2 Approches comportementalistes

Les approches *comportementalistes* (*behavior-based robotics*) reposent sur l'exploitation de plusieurs *comportements*, potentiellement combinés, que le robot peut prendre pour réaliser une tâche ou une mission donnée. Nous proposons ici deux exemples de travaux utilisant ce type d'approche.

L'architecture de commande HARPIC [Dalgalarrondo et al., 2004], développée par le centre DGA d'Arcueil, permet à une robot de reconnaissance d'intérieur de fonctionner sous des degrés différents d'autonomie, le degré le plus élevé étant assuré par une méthode comportementaliste. Les comportements possibles sont par exemple "longer un mur" ou "suivre une cible". Plusieurs types de représentation du monde sont possibles, le niveau de pertinence associé à ceux-ci dépendant naturellement du comportement. Les processus de perception qui doivent être activés ou inhibés sont ainsi fonction du type de représentation sélectionné, et les actions bas-niveau à effectuer sont choisies selon le comportement et la méthode de perception choisis.

[Payton et al., 1992] ont quant à eux développé un système de pilotage autonome tolérant aux fautes pour un véhicule sous-marin autonome. Cet engin opérant en eaux profondes, les

[18] les différentes couleurs possibles sont : *couloir, couloir avec amers, large porte, porte étroite, espace confiné, espace ouvert, espace ouvert avec caméras.*

possibilités de communication avec un opérateur humain sont très rares, il faut donc qu'il soit capable de trouver des solutions alternatives pour poursuivre sa mission en cas de problème tel qu'une faute inattendue.
La méthode développée porte le nom explicite de "Do whatever works" : si un comportement ne fonctionne pas comme souhaité (les actions semblent inefficaces car l'état du système n'évolue pas comme prévu), il faut trouver d'autres solutions, ce qui signifie utiliser d'autres comportements qui permettront d'atteindre tout de même les objectifs de la mission en cours. Pour cela, le véhicule comporte plusieurs niveaux de redondance, étant en particulier équipé de plusieurs actionneurs pour réaliser des opérations similaires (telles qu'augmenter l'altitude pour s'éloigner du fond de l'océan). Les comportements peuvent être combinés, lorsqu'ils ne mettent pas en jeu des actions contradictoires, en pondérant les actions qu'ils impliquent. Des lois simples permettent de savoir si un comportement donné doit être activé, à partir d'une fonction de besoin, un prédicteur de performance et des priorités de la mission en cours. Le prédicteur de performance d'un comportement doit fournir un "score d'échec" de ce dernier, à partir de fonctions simples.

1.4.3 Le projet EDEN du LAAS

Dans le groupe Robotique et Intelligence Artificielle (RIA) du LAAS, le projet EDEN (Expérimentations de Déplacements en Environnement Naturel) focalise et intègre les travaux du groupe en robotique d'intervention depuis 1993. L'objectif de ce projet interne est la conception, la réalisation et surtout l'intégration de méthodes et d'algorithmes pour la navigation autonome en environnement naturel.

Dès son origine, EDEN a proposé une approche *adaptative* de la navigation en environnement naturel, qui consiste à doter le robot de différents modes de génération et d'exécution des déplacements suivant le type de terrain rencontré [Chatila et al., 1995]. Cette approche est basée sur un principe d'économie des moyens, qui vise une meilleure efficacité : sur un type de terrain donné, le robot ne réalise que les actions de perception et de planification qui sont nécessaires. Au cours des années, divers modes de déplacement ont donc été proposés : différents modes de déplacement *réactifs*, applicables en terrain essentiellement plan et dégagé d'obstacles, et différents modes *planifiés*, activés lorsque le robot peut être amené à se sortir d'impasses ou à traverser des zones très accidentées.

Le choix du mode de déplacement à appliquer a jusqu'ici toujours été réalisé uniquement sur la base d'une représentation qualitative de l'environnement construite par le robot, qui explicite l'applicabilité des modes de déplacement possibles. Cette représentation est structurée en régions, et une recherche de chemin dans le graphe qu'elles définissent permet de sélectionner un sous-but à atteindre et le mode à appliquer [Lacroix et al., 1995].

Nos travaux se situent clairement dans ce contexte, en l'étendant grâce à un système de *supervision de la navigation*, qui évalue en ligne l'adéquation du mode de déplacement choisi.

1.5 Situations d'échec

Sur un système robotique autonome, de nombreux cas d'échecs peuvent se produire et il est d'autant plus crucial de savoir les repérer et les corriger que le robot a un degré élevé d'autonomie et qu'il est difficilement accessible par l'opérateur humain qui le supervise. Un des domaines dans lesquels cette constatation devient une évidence est l'exploration planétaire. En effet, encore aujourd'hui, lorsqu'un rover est en mission d'exploration de la surface de Mars, aucun humain ne se trouve sur place pour intervenir. De plus les délais de communication entre Mars et la Terre empêchent toute communication *réactive* avec les rovers. Ces derniers doivent donc disposer d'un

minimum d'autonomie, ne serait-ce que pour réaliser des missions d'une dizaine de minutes.

Les fautes qui peuvent intervenir sont de différentes catégories. On distingue souvent dans la littérature de diagnostic et FDIR[19] les fautes *matérielles* (panne de moteur, de caméra, casse mécanique, etc...) et *logicielles* (problème de gestion mémoire, "bug" tel qu'un problème de conversion d'unités). Ces *fautes* ou situations d'échec sont le plus souvent des événements "nets" tels que ceux définis précédemment.

On peut aussi rencontrer des situations dans lesquelles le comportement du système (par exemple la locomotion d'un rover) n'est pas "satisfaisante", c'est-à-dire que le comportement n'est pas celui attendu ou que le système n'atteint pas les performances souhaitées, avant bien souvent d'arriver dans une situation de *faute* effective. Dans ce cas, il peut être très intéressant d'utiliser des informations d'évaluation de ce comportement afin de réagir de la meilleure manière possible, permettant en particulier d'éviter une situation de blocage en *anticipant* les problèmes.

1.5.1 Un exemple : les malheurs d'Opportunity

Prenons pour exemple le cas du rover Opportunity, du programme MER. Opportunity reçoit chaque jour une petite mission consistant généralement à avancer de quelques mètres. Une fois la mission quotidienne terminée, les ingénieurs sur Terre étudient les données avant d'envoyer une nouvelle mission au rover. Il a ainsi parcouru quelques kilomètres depuis qu'il a commencé son parcours sur le sol martien (figure 1.10). Ayant jusque-là un parcours sans accroc particulier, le 26 avril 2004 (*Sol 446*) Opportunity s'est enlisé dans une dune de sable d'environ 30 centimètres (fig. 1.11), ne pouvant plus avancer, ni reculer (illustration parfaite de ce que nous appellerons une *faute de locomotion*). Il a alors fallu 5 semaines de travail et de manoeuvres aux ingénieurs de JPL pour le sortir de ce mauvais pas. N'ayant pas de retour suffisant sur les données internes du rover et surtout sur son comportement en ligne, cet incident n'a pas pu être anticipé, ce qui a eu d'importantes conséquences, même si l'histoire s'est bien terminée puisque le rover a pu repartir et continue encore son parcours depuis lors.

1.5.2 Leçons à tirer

Cet exemple illustre bien :

- l'utilité d'avoir **plusieurs modes de déplacement** pour évoluer dans de tels environnements avec la capacité de choisir le mode le mieux adapté. En effet, Opportunity n'a qu'un seul mode de locomotion, le *roulement simple*, or on peut raisonnablement penser qu'il aurait pu bénéficier d'un mode tel que le *péristaltisme* pour se déplacer plus efficacement et plus sûrement sur la dune de sable, terrain peu cohésif ;
- l'intérêt d'exploiter une **évaluation du comportement du mode de déplacement** courant du robot pour effectuer le choix de ce mode afin d'anticiper des situations de fautes et essayer si possible de les éviter.

Imaginons un rover tel qu'Opportunity dans la situation décrite précédemment, mais disposant d'un mode de péristaltisme et d'un tel système de choix de mode se servant des données de comportement. En arrivant sur cette dune *a priori* anodine avec un mode de locomotion de type roulement simple, des débuts de difficultés de locomotion apparaîtraient, pourraient être détectés par un processus de surveillance de comportement, amenant le rover à passer au mode péristaltisme *avant* d'être bloqué. Cela permettrait ainsi de franchir cette dune sans trop de

[19] Fault Detection, Identification and Recovery : i.e. Détection de fautes, identification et réparation

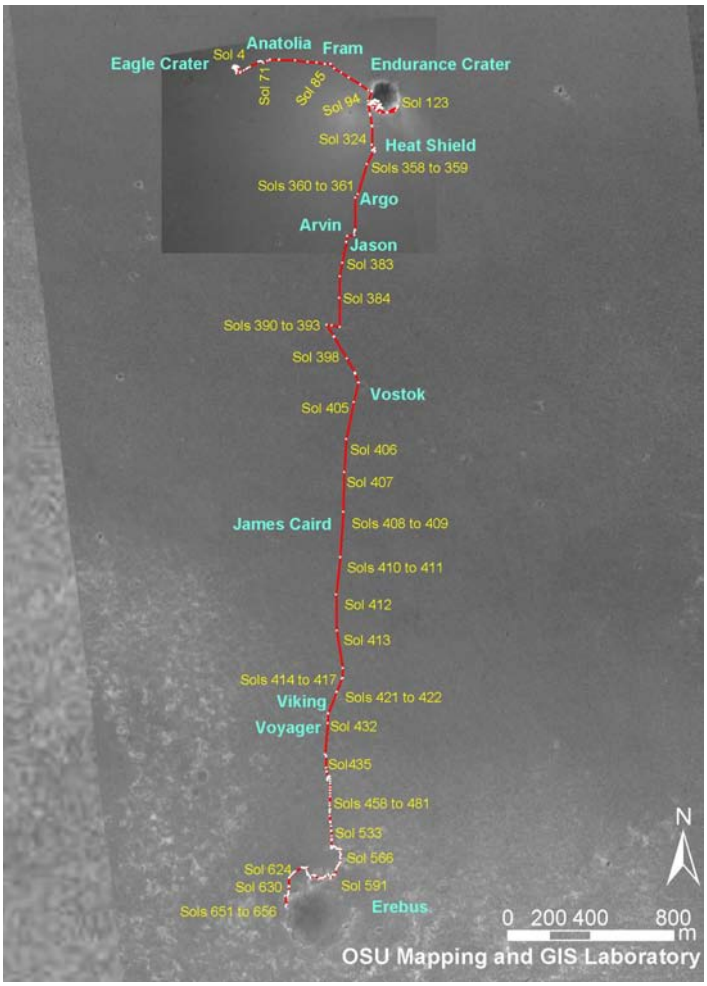


FIG. 1.10 – Carte du parcours réalisé par Opportunity jusqu'à Sol 656 (28 novembre 2005)

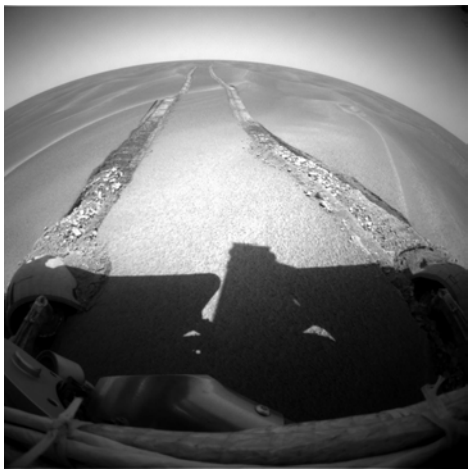


FIG. 1.11 – Traces des roues arrières du rover Opportunity après qu'il s'est enlisé dans une dune de sable (Sol 468)

difficulté pour revenir ensuite au mode de roulement d'origine (car bien plus rapide et plus économe en énergie que le péristaltisme).
La mise au point d'un tel système est précisément l'objet de cette thèse.

1.6 Conclusions

Nous avons pu voir l'intérêt que l'on peut tirer de la disponibilité de modes différents (et des fonctionnalités associées) pour la réussite d'une tâche de déplacement d'un robot mobile autonome en environnements naturels, consistant à rallier un point en une position spécifiée. Quelques exemples de composantes de ces modes possibles ont été présentés, en particulier des plates-formes disposant de différents modes de *locomotion* et des méthodes distinctes de représentation de l'environnement et de génération de mouvement (*navigation*). Nous avons également pu voir, au travers d'un exemple illustratif, combien il pouvait être utile et pertinent d'effectuer une évaluation du comportement du mode courant et de se servir de ces informations pour le choix du meilleur mode à appliquer.
Le premier problème désormais est la mise au point d'un système capable de sélectionner en ligne le meilleur mode à appliquer parmi ceux disponibles, dans le contexte difficile et très variable constitué par les environnements naturels, non structurés.
Le deuxième est la mise au point d'évaluateurs de comportement pour les modes.
Enfin, le troisième est la combinaison de ces deux composantes, avec en plus l'utilisation de données initiales liées au contexte.

La description formelle d'un tel système développé dans le cadre de cette thèse fait l'objet du prochain chapitre. La présentation des modes de déplacement effectivement exploités dans ces travaux sera proposée dans le chapitre 3, puis dans les chapitres 4 et 5 nous verrons comment sont générés les deux types de données exploitées : données dites *initiales* sur le terrain et données en ligne de comportement. Enfin, l'intégration complète et ses résultats seront discutés dans le dernier chapitre.

Chapitre 2

Formalisme de la sélection de mode de déplacement

Nous avons vu dans le chapitre précédent nos motivations pour l'utilisation d'une plate-forme robotique disposant de plusieurs types de *mode de déplacement*, à savoir des fonctionnalités de *navigation* (perception et mouvement prévu) et de *locomotion* (manière de réaliser le mouvement) permettant au robot d'effectuer une tâche de déplacement autonome. En effet, disposer de ces différentes possibilités peut permettre de faire face à une plus grande variété de situations, l'un des points délicats lorsque l'on veut faire évoluer un robot de manière autonome en environnement naturel a priori inconnu. Cette caractéristique peut également permettre d'éviter de se retrouver dans des situations compromises (comme nous l'avons vu pour Opportunity), à condition de faire le bon choix de mode au bon moment. Ce choix doit se faire en tenant compte naturellement du *contexte*, à savoir du type d'environnement dans lequel le robot se trouve, mais nous avons également pu entrevoir le bénéfice que l'on pouvait tirer de la prise en compte dans la sélection de mode d'une évaluation du *comportement* courant du robot et du mode actuellement utilisé.

Evoquer des problèmes d'estimation de comportement d'un système, et en particulier des cas d'*échec* ou de *faute*, amène naturellement à aborder le domaine du *diagnostic*, même si seule une partie de cette discipline nous concernera en pratique.

Nous avons choisi de traiter notre problème par une méthode probabiliste afin de prendre en compte les nombreuses incertitudes du monde de la robotique. Les techniques probabilistes sont désormais très couramment exploitées dans les applications de robotique mobile, pour la perception, le contrôle ou encore la décision.

Ce chapitre vise à introduire puis présenter les outils et le formalisme de l'approche que nous proposons pour cette sélection en ligne de mode, embarquée à bord de nos robots. Nous proposons dans la première section une introduction au diagnostic et un rappel de notions de probabilités et de systèmes markoviens qui sont utiles pour la mise en place de notre approche. Nous évoquons ensuite dans une deuxième section des travaux connexes, afin de positionner notre approche. Enfin, la troisième section de ce chapitre est consacrée à la présentation de cette approche : une méthode probabiliste pour une sélection réactive du mode à appliquer.

2.1 Notions utiles et contexte du formalisme

2.1.1 Introduction au diagnostic

Nous proposons une introduction au domaine du diagnostic, tout d'abord d'un point de vue général, avant de nous recentrer sur le diagnostic pour robot mobile en environnement naturel qui nous concerne plus particulièrement. Cette introduction est largement fondée sur le livre [Dubuisson, 2001], dont nous reprenons en particulier les définitions. Ainsi, selon la définition de cet ouvrage, le diagnostic "permet, à partir de l'observation d'un système, de prendre une décision permettant d'améliorer la conduite du procédé ou d'empêcher une dérive pouvant mener à des échecs".

Le diagnostic correspond au suivi du *mode de fonctionnement* d'un système. Un mode de fonctionnement peut se définir comme l'ensemble des états du système qui le placent dans une situation d'action (de fonctionnement) donnée.

Le suivi de mode de fonctionnement est un processus en trois étapes [Dubuisson, 2001] :

1. détection du mode de fonctionnement sous lequel le système se trouve,
2. identification et localisation de la cause d'un mauvais fonctionnement éventuel,
3. action sur le système découlant des deux étapes précédentes (maintien dans le même mode, correction du fonctionnement, arrêt si nécessaire).

Dans notre cas, nous nous intéresserons surtout à la *surveillance* du fonctionnement de la méthode de déplacement utilisée sur le robot (étape 1) et à l'action sur le système pour changer de méthode le cas échéant (étape 3), particulièrement lorsqu'un mauvais fonctionnement a été détecté.

Le *diagnostic par analyse de signatures* (chapitre 1 de [Dubuisson, 2001]) nous intéresse plus particulièrement. Il est parfois apparenté au diagnostic de maladies dans le domaine médical, dans lequel les *signatures* sont les *symptômes* de la maladie. En effet, à partir de l'observation des symptômes on déduit la nature des maladies, ce qui constitue un diagnostic. Ainsi, le diagnostic par analyse de signatures externes se réalise généralement en quatre étapes (figure 2.1), les trois premières concernant la *détection de fautes* et la quatrième l'*identification* :

1. Acquisition des signatures. Les signatures (symptômes) peuvent être obtenues par les sens humains ou des capteurs, avec ou sans stimulation externe et être statiques ou dynamiques.
2. Comparaison des signatures avec les signatures de référence. Il s'agit de vérifier que l'information obtenue à un instant donné reste conforme aux normes de fonctionnement dans les limites connues de tolérance : on a donc besoin en général de banques de données, de normes, de seuils admissibles. Les *signatures de référence* décrivent le comportement *normal* attendu du système. Elles peuvent être qualitatives ou quantitatives.
3. Phase de décision pour déclarer la signature normale ou anormale. En prenant en compte l'incertitude, l'objectif est de classer le fonctionnement en trois grandes catégories : fonctionnement *normal*, *anormal connu* et *anormal inconnu*. Pour cela, on définit des seuils permettant d'accepter avec un risque raisonnable les non-détections et fausses alarmes.
4. Interprétation de la signature à des fins de diagnostic. Cette étape est déclenchée quand le fonctionnement est considéré comme *anormal* (signature différente de la signature de référence).

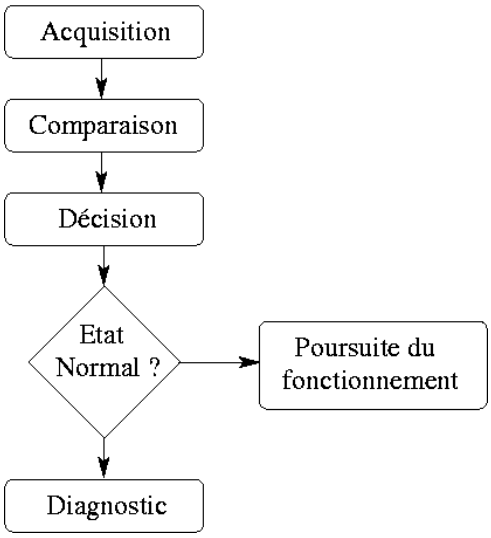


FIG. 2.1 – Procédure complète de diagnostic, d'après [Dubuisson, 2001]

Génération des signatures

La génération des signatures utilisées dans les étapes du diagnostic exploite deux types de redondances :

- La *redondance matérielle* (ou de mesures) : on dispose de plusieurs capteurs permettant de mesurer la même grandeur et on compare les informations fournies par ces différents capteurs.
- La *redondance analytique* : on ne dispose pas de redondance matérielle "directe" mais on estime une grandeur de plusieurs manières différentes. On effectue une comparaison entre la mesure directe et une (ou des) déduction(s) à partir d'autres variables mesurées. Ce type de redondance est aussi parfois appelé *redondance par modèle*.

On peut également comparer des signatures obtenues à partir de mesures ou d'estimations de grandeurs utiles avec une référence de comportement (par exemple un *modèle de comportement nominal* ou attendu).
Le *vecteur de parité*, ou vecteur de résidus, est une sorte d'indicateur des défaillances qui est issu précisément de la comparaison des signatures courantes avec les signatures de référence. Il est très proche de zéro en fonctionnement nominal et significativement non nul en présence de défaillances.

Démarche classique de diagnostic

On dispose d'un modèle décrivant le fonctionnement normal du système et on surveille le fonctionnement réel en testant la cohérence entre ce modèle et les observations par comparaison de signatures. Si ces dernières ne vérifient pas la description du modèle, on en déduit que le fonctionnement réel n'est pas le fonctionnement nominal, déclenchant alors une alarme (phase de *détection*). Dans une deuxième phase, d'éventuels modèles de mauvais fonctionnement sont utilisés selon le même principe pour déterminer les défaillances présentes (étape de *localisation* des défaillances). Cela nécessite de construire des *modèles testables*, i.e. ne faisant intervenir que des variables disponibles, mesurées ou connues à l'avance. L'existence de ces modèles est

conditionnée par celle d'une redondance d'informations sur le système.

Modèles numériques de comportement La formulation de modèles numériques de comportement peut être effectuée dans le domaine temporel ou symbolique (voire les deux). Le plus souvent, les modèles de bon et de mauvais fonctionnement sont intégrés dans une formulation unique faisant apparaître des variables de défaillances $\boldsymbol{\delta}$: elles sont nulles en fonctionnement normal, non nulles en présence de défaillances. Différentes causes de défaillance correspondent à différentes composantes du vecteur $\boldsymbol{\delta}$. Le problème de *détection* est de reconnaître la non-nullité de $\boldsymbol{\delta}$. Le problème de *localisation* (ou identification) est de reconnaître quelles composantes sont non nulles.

2.1.2 Rappels de probabilités

Après un aperçu des motivations pour l'utilisation de la théorie des probabilités en robotique, nous proposons un bref rappel de quelques éléments qui seront utiles dans notre application.

La robotique probabiliste

Les systèmes robotiques évoluent dans le monde physique, perçoivent des informations sur leur environnement grâce à des capteurs et agissent pour réaliser leurs tâches. Pour cela, ils doivent être capables de gérer les fortes incertitudes auxquelles tout système percevant et agissant dans le monde physique réel doit faire face. Les facteurs contribuant à l'incertitude pour un robot sont multiples [Thrun et al., 2005] :

- Les environnements robotiques sont fondamentalement imprévisibles, particulièrement lorsqu'ils sont non structurés et/ou dynamiques.
- Les capteurs comportent des limitations dans leur perception (de résolution ou de champ de vue par exemple). Ils sont sujets au bruit qui perturbe les mesures de manière imprévisible, limitant l'information pertinente qui peut en être extraite. Enfin, ils peuvent être sujets à des pannes.
- Les actionneurs des robots sont également imprévisibles dans une certaine mesure : l'incertitude peut provenir du bruit dans la commande, de défaillances mécaniques etc...
- La partie logicielle d'un robot peut également comporter de l'incertitude. Les *modèles* utilisés pour représenter le robot ou le monde sont des abstractions du monde réel : ils sont approximés. Les erreurs de modèle peuvent ainsi générer des incertitudes significatives.
- Une autre source d'incertitude sont les "approximations algorithmiques". En effet, les robots se doivent d'être des systèmes *temps réel*, ce qui limite la quantité de calculs qui peuvent être réalisés à bord, amenant souvent à effectuer des approximations simplificatrices lors de l'intégration sur un robot. De plus la conception même de l'informatique à bord fait que les données évoluent en pratique à temps discret.

Le niveau d'incertitude finalement atteint dans une application robotique peut être très élevé, ce qui fait de sa gestion un des problèmes majeurs de la robotique. La *robotique probabiliste* permet de représenter explicitement l'incertitude en exploitant la théorie des probabilités et la puissance de son formalisme mathématique. Elle permet ainsi de traiter des problèmes de perception, contrôle, planification et décision en combinant judicieusement les données incertaines. D'autres formalismes tels que les méthodes ensemblistes, la logique floue ou la théorie des croyances de Dempster-Shafer [Bloch, 1996] proposent de traiter certains de ces problèmes de manière différente, mais ils ne disposent pas (encore) de la même expertise d'une large part

de la communauté robotique. Pour ces raisons nous avons choisi de réaliser notre application en utilisant le formalisme des probabilités.

Notations

Tout d'abord, nous distinguerons les **vecteurs** comme $\boldsymbol{X}$ des variables comme X en les faisant apparaître en caractère gras.

Considérons une variable aléatoire X, nous noterons X_t l'état de X à l'instant t et $X_{0:t}$ l'ensemble des X_k pour tout temps k compris entre 0 et t, i.e. : $X_{0:t} = \{X_0, X_1, \ldots, X_{t-1}, X_t\}$. Le vecteur des observations au temps t sera quant à lui noté, avec la même convention : $\boldsymbol{O}_t$.

La probabilité conditionnelle de la *conjonction* de X et Y est notée $P(X, Y) = P(X \wedge Y)$.

La probabilité de l'événement X *sachant* Y est notée : $P(X|Y)$.

Règle de Bayes

La *règle de Bayes* lie une probabilité conditionnelle telle que $P(X|Y)$ à son "inverse", $P(Y|X)$:

$$P(X, Y) = P(X|Y)\, P(Y) = P(Y|X)\, P(X)$$

Cette règle joue un rôle fondamental dans l'estimation probabiliste, et en particulier dans la robotique probabiliste. Elle permet d'inférer X à partir de données Y (généralement des observations). On estime ainsi une probabilité dite *a posteriori* $P(X|Y)$ à partir de la probabilité conditionnelle "inverse" $P(Y|X)$ qu'apportent les données et d'une probabilité dite *a priori* $P(X)$. Ainsi, dans le cas discret, la probabilité *a posteriori* de l'état X s'écrit :

$$P(X|Y) = \frac{P(Y|X)\, P(X)}{P(Y)} = \frac{P(Y|X)\, P(X)}{\sum_{X'} P(Y|X')P(X')} \tag{2.1}$$

Normalisation

Afin de simplifier les expressions telles que la précédente nous noterons en général η la constante de normalisation nécessaire pour assurer que la somme des probabilités est égale à 1 (contrainte d'additivité des probabilités). Elle apparaîtra généralement dans l'application de la règle de Bayes, c'est pourquoi elle est parfois appelée *variable de normalisation dans la règle de Bayes*. Nous pourrons ainsi noter η la quantité $P(Y)^{-1}$ dans l'équation (2.1), ce qui nous donne la formulation simplifiée :

$$P(X|Y) = \eta\, P(Y|X)\, P(X)$$

Précisons que nous nous permettrons d'utiliser indifféremment cette notation η comme symbole de normalisation pour différentes équations probabilistes, même si la valeur effective de η n'est pas la même en pratique.

Indépendance de deux variables aléatoires

L'*indépendance* (absolue) entre deux variables aléatoires X et Y se traduit par l'égalité suivante sur leurs probabilités : $P(X, Y) = P(X)P(Y)$.
Notons que l'*indépendance* de X et Y implique que : $P(X|Y) = P(X)$.
Bien entendu, de la même manière : $P(Y|X) = P(Y)$.

L'*indépendance conditionnelle* entre X et Y vis à vis de Z s'applique lorsque Y ne contient pas d'information sur X si Z est connue. Elle peut se traduire par les égalités suivantes : $P(X,Y|Z) = P(X|Z)P(Y|Z)$ ou encore :

$$P(X|Z,Y) = P(X|Z) \tag{2.2}$$
$$P(Y|Z,X) = P(Y|Z)$$

Notons que l'indépendance conditionnelle *n'implique pas* l'indépendance absolue.

$$P(X,Y|Z) = P(X|Z)P(Y|Z) \quad \not\Rightarrow \quad P(X,Y) = P(X)P(Y)$$

La réciproque est également fausse en général.

$$P(X,Y) = P(X)P(Y) \quad \not\Rightarrow \quad P(X,Y|Z) = P(X|Z)P(Y|Z)$$

2.1.3 Modèles de Markov cachés

Un modèle de Markov caché est un modèle temporel probabiliste, dérivé des chaînes de Markov. Les chaînes de Markov représentent des processus de Markov discrets par un automate

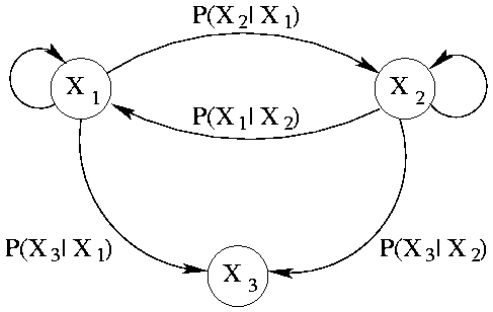


FIG. 2.2 – Un exemple d'une chaîne de Markov à trois états avec ses transitions probabilistes. L'état X_3 est ici un état "puits" : il n'y a pas d'arc de transition sortant, car toutes les probabilités de transition vers des états différents sont nulles.

stochastique à état fini, où les états sont représentés par des noeuds et les transitions possibles entre ces états sont symbolisées par des arcs orientés, auquels sont associés des probabilités de transition (les transitions sont donc probabilistes, voir figure 2.2). La somme de l'ensemble des probabilités des transitions sortantes d'un noeud (état) donné doit être égale à 1. Les processus discrets de Markov vérifient l'hypothèse de Markov : *l'état présent ne dépend que d'un nombre fini d'états passés* [Russell et al., 2003a]. Elle s'exprime ainsi de la façon suivante à l'ordre k :

$$P(X_t|X_{0:t-1}) = P(X_t|X_{t-1}, X_{t-2}, \ldots, X_{t-k})$$

A l'ordre 1, cette hypothèse peut se traduire par la formulation suivante : l'état présent ne dépend que de l'état précédent :

$$P(X_t|X_{0:t-1}) = P(X_t|X_{t-1})$$

En d'autres termes, connaissant l'état courant, le futur est indépendant du passé.
Si cette hypothèse est aussi valable pour les variables d'observation, les O_t ne dépendent que des observations à l'état courant X_t :

$$P(O_t|X_{0:t}, O_{0:t-1}) = P(O_t|X_t) \tag{2.3}$$

Un modèle de Markov caché (HMM, pour Hidden Markov Model) est une chaîne de Markov d'ordre 1 (i.e. respectant l'hypothèse de Markov à l'ordre 1 sur les états et les observations) qui comporte des états non observables, dans laquelle l'état du procédé est représenté par *une* variable aléatoire discrète. Les valeurs possibles de cette variable sont les états possibles du monde [Russell et al., 2003b].

Pour exemple, dans un contexte de diagnostic, considérons un système pouvant être dans un des modes $\{standby, on, fault\}$ [da Silveira et al., 2005]. L'observation peut permettre de faire la distinction entre un mode de fonctionnement normal ($\{standby\}$ ou $\{on\}$) et le mode de faute $\{fault\}$ mais ne pourra pas faire la distinction entre $\{standby\}$ et $\{on\}$. Le modèle de Markov correspondant est donc *caché*.

Une description complète d'un HMM doit comporter :
- l'ensemble des états possibles,
- les observations O pouvant être obtenues pour chaque état,
- le *modèle d'observation*, donnant les densités de probabilité d'obtenir une observation sachant l'état : $P(O|X)$,
- le *modèle dynamique*, donnant les probabilités de transitions entre états : $P(X|X')$,
- la distribution initiale des états (probabilités *a priori* $P(X)$ de tous les états).

2.2 Positionnement bibliographique

Si l'on peut voir dans la littérature un nombre croissant de plates-formes robotiques d'extérieur capables de se déplacer selon différents modes de locomotion ou de navigation (plusieurs exemples ont été succinctement présentés dans le chapitre 1), il existe à notre connaissance très peu de systèmes capables d'estimer de manière autonome le mode à appliquer, pour le sélectionner effectivement. Néanmoins, depuis quelques années, la communauté robotique s'intéresse à :
- l'exploitation des méthodes de diagnostic pour la robotique mobile, proposant ainsi des estimations de *modes de fonctionnement* permettant la détection de situations de fautes ou d'échecs,
- la mise en oeuvre des systèmes de décision reposant sur un formalisme probabiliste pour le contrôle de rovers autonomes, allant vers le choix autonome d'actions à réaliser telles qu'un déplacement ou une prise d'image.

Nous décrivons quelques-uns de ces travaux qui nous semblent intéressants vis-à-vis de notre problème. Nous verrons quelles sont leurs contributions, et comment nous situons nos propres travaux vis-à-vis de ces derniers.

2.2.1 Diagnostic en robotique mobile autonome

Le diagnostic pour robots mobiles autonomes en environnement naturel (il s'agit le plus souvent de *rovers*) a vu de nombreux développements ces dernières années, prenant une importance croissante avec l'augmentation de l'autonomie et de la complexité de ces véhicules. L'une des principales contributions en la matière est constituée par les travaux de Brian C. Williams et ses collaborateurs du MIT[1] sur le diagnostic dit *"basé modèle"*[2]. Il étudie des systèmes *hybrides* avec transitions de modes *autonomes*. L'état est $\boldsymbol{x} = (x_d, \boldsymbol{x_c})$, où x_d est une variable discrète,

[1] Massachusetts Institute of Technology, Cambridge, MA, USA
[2] *model-based* diagnosis

prenant un nombre fini de valeurs, qui représente le *mode opérationnel* du système, et $\boldsymbol{x_c}$ est l'*état continu*. Parmi les *modes opérationnels* du système, on trouve des modes de fonctionnement normal et des modes d'échec (fautes).

Automate Hybride Probabiliste

Les auteurs de [Funiak et al., 2003] prennent un exemple simple pour illustrer le propos : celui d'un robot jongleur à deux degrés de liberté. Ce système peut être dans quatre modes : $\{m_{0,ok}; m_{1,ok}; m_{0,failed}; m_{1,failed}\}$, correspondant au fait que le robot tient ou non une balle $\{m_0, m_1\}$, et au fait que l'actionneur est en échec ou non $\{failed, ok\}$. La capture de la balle par le robot est ici un exemple de transition de mode autonome : les probabilités de transition dépendent de l'*état continu*.
Afin de réaliser le diagnostic de ce système hybride, les auteurs utilisent un *Automate Hybride Probabiliste* (PHA : *Probabilistic Hybrid Automaton*), un formalisme qui combine les modèles de Markov cachés (HMM) et les modèles de systèmes dynamiques continus (tels que le filtre de Kalman). Ce formalisme est décrit par :

- l'*état hybride*,
- un ensemble de variables d'entrée/sortie (actions discrètes $\boldsymbol{u_d}$, commandes continues $\boldsymbol{u_c}$ et sorties continues $\boldsymbol{y_c}$),
- une fonction de *transition discrète* spécifiant une distribution sur les modes à l'instant t à partir des valeurs possibles de $\boldsymbol{u_d}, \boldsymbol{u_c}, x_d, \boldsymbol{x_c}$,
- une *évolution continue* de l'automate pour chaque mode x_d (constituée d'une fonction de transition, d'une fonction d'observation et de bruits gaussiens indépendants du mode).

Les dépendances conditionnelles entre ces différentes variables sont illustrées dans la figure 2.3.

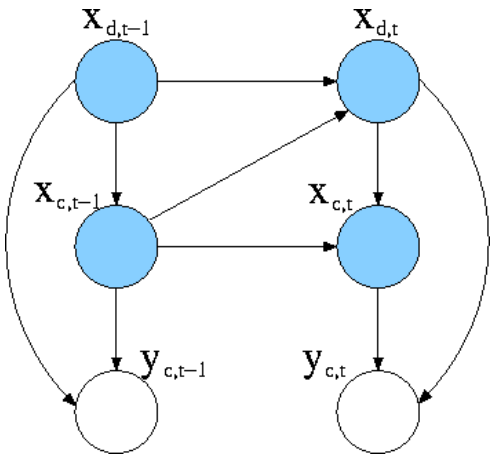


FIG. 2.3 – D'après [Funiak et al., 2003], dépendances conditionnelles entre les variables d'état $\boldsymbol{x_c}$, x_d et l'observation $\boldsymbol{y_c}$, représentées sous la forme d'un réseau bayésien dynamique (DBN, voir [Russell et al., 2003a]). L'arc de $\boldsymbol{x}_{c,t-1}$ à $x_{d,t}$ représente la dépendance de ce dernier sur le premier, soit les-dites *transitions autonomes*.

L'objectif principal d'une telle application de diagnostic est d'estimer le mode de fonctionnement courant du système afin d'identifier en particulier les états de faute (état *failed* par exemple pour le robot acrobate). L'algorithme exploité dans ces travaux pour effectuer l'estimation du mode (ainsi que de l'état continu) exploite une combinaison de filtrage particulaire avec une

représentation gaussienne multi-modale.

Le filtrage particulaire pour le diagnostic

Le fonctionnement du filtrage particulaire est le suivant [Russell et al., 2003a] : une population de N particules est créée en échantillonnant (discrétisant) la distribution de probabilité initiale, à $t = 0$. Puis le cycle suivant est répété à chaque échantillon de temps :

- Chaque particule est propagée vers le futur en échantillonnant la valeur du prochain état $\boldsymbol{x}_{t+1}$, sachant la valeur courante $\boldsymbol{x}_t$ pour la particule et en utilisant le modèle de transition $P(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)$.
- A chaque particule est associé un *poids* (ou une pondération) en fonction de la *vraisemblance* que la nouvelle observation lui attribue : $P(\boldsymbol{O}_{t+1}|\boldsymbol{x}_{t+1})$.
- La population de particules est *ré-échantillonnée* pour générer une nouvelle population de N particules. Chaque nouvelle particule est choisie dans la population actuelle ; la probabilité qu'une particule donnée soit sélectionnée est proportionnelle à son poids. Ces nouvelles particules ne comportent plus de poids.

Le *filtrage particulaire* est de plus en plus utilisé pour des problèmes d'estimation non paramétrique et non linéaire, dans des applications de plus en plus diverses, telles que le SLAM [Thrun et al., 2005] ou le suivi visuel [Brèthes, 2005]. Ses principaux attraits sont la possibilité de traiter des distributions non paramétriques et le fait que le coût en temps de calcul dépend seulement du nombre de particules, et non pas de la complexité du modèle. Il est néanmoins bien moins adapté (à l'origine) au problème du diagnostic et de la détection de fautes du fait que les transitions de mode les plus intéressantes dans ces domaines, à savoir celles vers les états de fautes, sont précisément celles qui ont la probabilité d'occurrence la plus faible. Ceci a pour conséquence un risque qu'il n'y ait plus de particule dans un état de faute quand cette faute se produit, rendant le système incapable de la détecter. Cependant, [Dearden et al., 2002] ou [Verma et al., 2001] ont proposé des méthodes permettant d'éviter ce problème, en garantissant qu'il y a toujours assez de particules dans les états "intéressants". Ces méthodes ont permis de rendre le filtrage particulaire efficace aussi pour ces problèmes de diagnostic.
Dans [Dearden et al., 2002], les auteurs effectuent ainsi un *échantillonnage suivant l'importance* ("importance sampling") : au lieu d'effectuer l'échantillonnage sur la distribution P qui est la probabilité a posteriori que l'on cherche à calculer, ils échantillonnent sur une autre distribution Q en pondérant chaque particule s par le rapport de probabilités $P_P(s)/P_Q(s)$. La distribution Q est construite grâce à un "oracle" capable de fournir un ensemble d'états candidats dans lesquels le système peut se retrouver à partir de la distribution actuelle, et en faisant en sorte qu'il y ait toujours des particules dans ces états.
[Verma et al., 2001] effectuent quant à eux une généralisation du filtrage particulaire à partir des notions de théorie de la décision, en s'inspirant de l'intervention dans cette dernière d'un critère de *coût* (ou de *récompense*) pour une action. Ainsi, l'importance relative des états est représentée par une fontion d'*utilité* $u(X)$ qui intervient dans la "re-pondération" des particules par l'intermédiaire du rapport $u(x_{t+1})/u(x_t)$. Les étapes d'échantillonnage, re-pondération et ré-échantillonnage sont en fait effectuées en une étape en utilisant une probabilité de transition qui contient la re-pondération : $(u(x_{t+1})/u(x_t))\, p(x_{t+1}|x_t)$.

Automate Probabiliste Hiérarchique

Une version hiérarchisée du PHA est évoquée dans [Mikaelian et al., 2005] avec le PHCA[3], "automate probabiliste hiérarchique basé contraintes", présenté comme un codage compact des HMM pouvant permettre de représenter à la fois le comportement matériel (*hardware*) et logiciel (*software*) pour diagnostiquer des fautes dans l'un ou l'autre de ces domaines en utilisant le même modèle. Ce formalisme représente le système en "macro-modes" de fonctionnement qui eux-mêmes peuvent contenir des modes "primitifs" tels que des états de fautes précis (voir l'exemple de la figure 2.4).

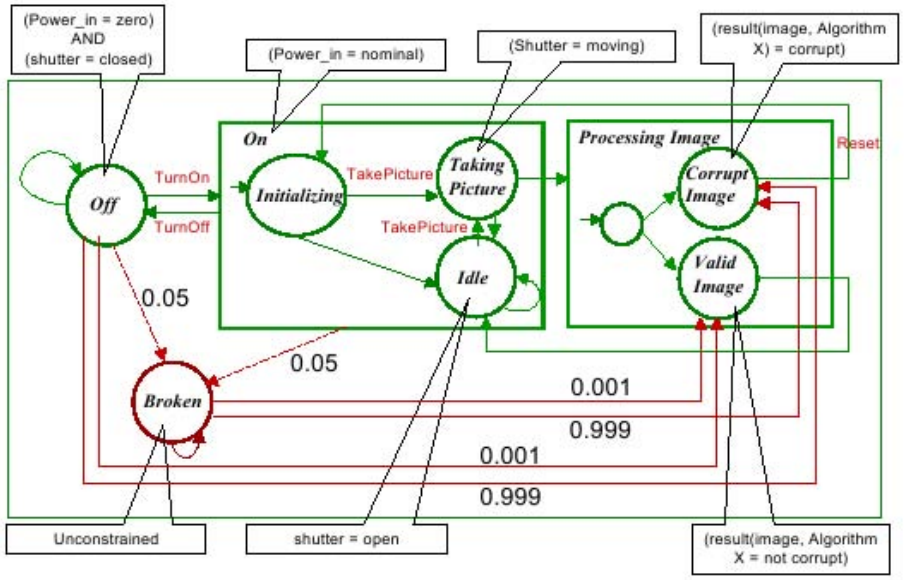


FIG. 2.4 – Exemple d'un modèle PHCA (*Probabilistic Hierarchical Constraint-based Automata*) pour un module caméra/traitement d'image, d'après [Mikaelian et al., 2005]. Les ronds sont des modes "primitifs" et les rectangles les englobant sont des "macro-modes".

Remarques

Notons que même dans ces systèmes de diagnostic, les transitions de modes sont *fixées a priori*, généralement à partir de l'expérience du concepteur (ou développeur). Dans le cas du PHCA nous voyons plus précisément des transitions *déterministes* guidées par les actions réalisées (telles que le `TakePicture` de la figure 2.4) et des transitions à probabilités fixées (typiquement, celles vers les états de fautes).

Les fautes sont des états discrets précis et qualitatifs que l'on cherche à identifier. Ces systèmes ne s'intéressent pas à effectuer une évaluation du comportement dans le temps qui soit plus *quantitative*, se servant de *tendances* vers l'échec pour éventuellement *agir* en conséquence et/ou éviter les problèmes qui semblent sur le point d'apparaître. Cela est dû en partie à la nature des *fautes* ou *échecs* que l'on cherche à détecter. Ce sont surtout des fautes *majeures* telles que `caméra en panne`, `roue cassée` (pour la partie hardware) ou `bug logiciel` (pour la partie software).

[3] *Probabilistic Hierarchical Constraint-based Automata*

Ces systèmes sont passifs, dans le sens où aucune action particulière n'est déclenchée par le système (ou même prévue) en cas de détection de faute.

2.2.2 Contrôle actif d'un rover par MDP

D'autres systèmes plus actifs s'intéressent à définir une politique d'*actions* pour un rover. Dans [Bernstein et al., 2001], un système de contrôle actif construit sous la forme d'un procédé décisionnel de Markov (MDP[4]) est proposé pour piloter un rover planétaire ayant pour mission de récolter des informations scientifiques (prises d'images, analyse au spectromètre).
Un MDP est un agent qui agit dans un environnement stochastique pour maximiser une *récompense* à long terme. Il est généralement utilisé pour des problèmes de *planification*. A chaque instant t (nous considérons un temps discret), l'agent perçoit et estime l'état de l'environnement x_t et choisit une action a_t parmi l'ensemble des actions possibles. Au temps suivant $(t+1)$, l'environnement produit une *récompense* numérique r_t et un état suivant x_{t+1}. La *dynamique* de l'environnement est modélisée par des probabilités état-transition $P(x, a, x')$, également dépendant de l'action[5], et des récompenses immédiates attendues : $R(x, a)$. Une *politique* est une correspondance états-actions : elle définit quelle action réaliser dans un état donné. Une fois le MDP construit, l'objectif est de trouver une politique qui maximise la récompense à long terme. Cette récompense peut par exemple être calculée en sommant toutes les récompenses jusqu'à la fin de l'*épisode* (ou mission) si celle-ci est connue, ou sinon en sommant sur un épisode de longueur finie [Bernstein et al., 2001].
Une des difficultés des MDP est de construire le *modèle dynamique*, à savoir générer les probabilités de transition d'états et les récompenses associées aux états courants. Ceci est souvent réalisé à l'aide de techniques d'apprentissage telles que l'*apprentissage par renforcement* [Russell et al., 2003c]. Afin d'appliquer une telle technique, deux options sont envisageables :

- On peut faire apprendre son environnement au robot en le faisant longuement parcourir ce dernier. Ceci peut être réalisable pour des environnements d'intérieur (tels qu'un musée ou un hall d'exposition par exemple, voir [Morisset, 2002] et section 1.4.1) mais est bien plus délicat pour des environnements naturels vu la diversité des situations qui y est encore largement supérieure, voire pratiquement impossible pour des rovers planétaires. D'autre part, vu la faible probabilité des fautes, cela signifierait effectuer un nombre considérable de tests dans des conditions multiples pour pouvoir espérer obtenir des résultats valables, rendant cette option peu réalisable.
- On peut aussi envisager utiliser un *simulateur* pour construire le modèle (c'est le cas dans [Bernstein et al., 2001]). Cependant, réaliser un simulateur réaliste et suffisamment complet reste un problème très délicat.

Malgré le recours à des techniques d'apprentissage, la construction de ce modèle nécessite encore une bonne part d'*ingéniérie de connaissance*, ou expertise *a priori*, pour régler les paramètres à ajuster et effectuer les décisions de conception. En pratique, ces probabilités de transition sont le plus souvent *choisies* et *fixées* à partir de l'expérience des utilisateurs du robot [Fernández, 2000].

D'autre part, la lourdeur des modèles et du calcul de politique optimale amène souvent les auteurs à calculer le modèle du MDP et la politique *hors-ligne*.
Notons que dans le cas où l'état n'est pas directement observable, on parle de procédé décisionnel de Markov *partiellement observable* (POMDP[6]). Dans ce cas, les observations fournissent des

[4] Markov Decision Process

[5] elle peut aussi s'écrire sous la forme $P(x'|a, x)$, probabilité de l'état suivant x' sachant l'état courant x et l'action a.

[6] Partially Observable Markov Decision Process

informations partielles sur l'agent puisque la même observation peut être réalisée dans différents états. On se rapproche en ce sens de la notion de HMM, avec en plus les *actions* telles qu'elles sont formulées dans les MDP. [Verma et al., 2002] utilise ainsi un POMDP pour effectuer du *monitoring* (surveillance) et de la détection de fautes pour un robot d'intérieur.

2.2.3 Bilan

Les approches existantes que nous pouvons rapprocher de nos objectifs présentent les caractéristiques suivantes :

1. Elles réalisent une détection de fautes "nettes" de type matérielles (e.g. casse moteur) ou logicielles (bug), sans effectuer une véritable *évaluation* du *comportement* d'un mode (même si une casse de ce type peut avoir des conséquences évidentes sur le fonctionnement d'un mode).
2. Les systèmes de diagnostic pour les rovers sont généralement passifs : quand une faute est détectée, le système est arrêté ou nécessite un *rétablissement* complet.
3. Les systèmes de contrôle actif des rovers (souvent construits comme des MDP) effectuent une *planification* à plus ou moins long terme, cherchant une politique optimale qui maximise une récompense.
4. La *dynamique* du modèle est construite soit en *fixant a priori* les probabilités de transition à partir de l'expérience du concepteur, soit en réalisant un *apprentissage*. Cette dernière option se révèle en pratique délicate et coûteuse. Elle n'est réalisée en général que pour des applications de robotique d'intérieur, du fait de la trop grande diversité de situations rencontrées en environnements naturels et le manque de possibilités de faire des hypothèses sur cet environnement.

Nous proposons dans la section suivante notre approche pour la sélection de mode de déplacement, que nous positionnerons en préambule vis-à-vis de ce bilan bibliographique.

2.3 Approche proposée

Notre objectif est de réaliser un système effectuant en ligne la sélection du *mode de déplacement* (navigation et locomotion) à appliquer sur un robot mobile d'extérieur qui a pour mission d'atteindre un point objectif dont les coordonnées (x, y) lui ont été spécifiées.
Ce choix de mode sera effectué sur la base de deux notions :

- le *contexte*, soit l'observation de la situation présente et son exploitation en utilisant des connaissances *a priori* sur les modes (contexte dédié ou nécessaire),
- le *comportement* du mode de déplacement en cours d'utilisation, reposant sur une évaluation de son fonctionnement ; l'idée étant qu'un mauvais comportement provoque une transition vers un autre mode applicable, afin d'améliorer le déplacement du robot et même de le poursuivre autant que possible.

Par rapport aux approches connexes proposées dans la littérature et précédemment présentées, nous souhaitons mettre en oeuvre un système présentant les caractéristiques suivantes :

1. Prise en compte de manière *quantitative* du *comportement* du robot, et plus particulièrement des modes de déplacement existants.
2. Cette étude de *comportement*, en plus de celle du *contexte*, doit mener à une sélection de mode donc à une conséquence *active*.

3. Notre système doit être *réactif* et avoir un coût de calcul faible pour fonctionner entièrement en ligne et à une fréquence potentiellement élevée (par rapport à un planificateur par exemple).
4. Il doit prendre en compte le comportement réel et actuel du robot (lorsque ces informations sont disponibles) pour adapter la *dynamique* de transition de modes, en combinant ces informations aux données classiquement fixées a priori.

Notons que notre système n'est pas destiné à *remplacer* un système de détection de fautes graves de type casse moteur car il n'effectuera pas l'identification de telles fautes, même s'il peut lui-même révéler certaines anomalies. Il gagnera au contraire à être juxtaposé à des procédés de détection de fautes tels que ceux qui viennent d'être évoqués.

Méthode proposée

Nous posons le problème comme un problème d'*estimation*, dans lequel les *états* sont les *modes de déplacement à appliquer. Nous ne cherchons pas à estimer le mode dans lequel se trouve le robot*, ceci étant parfaitement connu puisqu'imposé par notre système, mais plutôt à estimer quel mode devrait être appliqué à l'instant présent. Ceci se fera en utilisant un Modèle de Markov Caché (HMM) : il s'agit bien d'un modèle caché car l'état n'est clairement pas directement observable. Ce HMM sera alimenté principalement par deux types d'information : des données sur le type de terrain parcouru, dites de *contexte*, et des données sur le *comportement* du robot, et plus particulièrement celui du mode actif (figure 2.5).

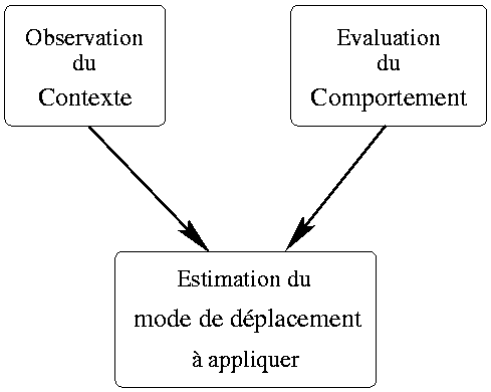


FIG. 2.5 – L'estimation du mode de déplacement à appliquer est effectuée grâce aux données de *contexte* et de *comportement*.

Notation : L'état du modèle de Markov caché sera noté x, prenant ses valeurs dans l'ensemble des états possibles $\{x_k\}_{k\in[\![1,N]\!]}$, avec $N =$ nombre de modes. Les différents modes de déplacement disponibles seront quant à eux notés $\{m_k\}_{k\in[\![1,N]\!]}$. Etre dans l'état x_k signifie donc que le mode à appliquer est le mode m_k. La probabilité que le mode m_k soit celui à appliquer à l'instant t sera ainsi notée : $P(x_{k,t})$.

2.3.1 Formulation "classique" d'un problème d'estimation

Connaissant $P(X_{t-1}|O_{1:t-1})$, une estimation de l'état au temps $(t-1)$ sachant toutes les observations jusqu'à $(t-1)$, on cherche à estimer $P(X_t|O_{1:t})$, à savoir l'estimation de l'état au

temps suivant, X_t, sachant que l'on a obtenu la nouvelle observation O_t.
Exprimons cette probabilité :

$$\begin{aligned} P(X_t|O_{1:t}) &= P(X_t|O_{1:t-1}, O_t) \\ &\qquad \text{(en séparant les observations)} \\ &= \eta\, P(O_t|X_t, O_{1:t-1}) P(X_t|O_{1:t-1}) \\ &\qquad \text{(règle de Bayes)} \\ &= \eta\, P(O_t|X_t) P(X_t|O_{1:t-1}) \\ &\qquad \text{(propriété de Markov sur les observations)} \end{aligned} \tag{2.4}$$

Ainsi :

$$P(X_t|O_{1:t}) = \eta\, P(O_t|X_t) P(X_t|O_{1:t-1}) \tag{2.5}$$

Cette méthode est parfois nommée *estimation récursive* [Russell et al., 2003b].

On retrouve ici les deux étapes du processus d'estimation. Connaissant $P(X_{t-1}|O_{1:t-1})$:

1. L'étape de *prédiction* effectue une première estimation de l'état suivant en prenant en compte le modèle *dynamique*, et en fonction de toutes les informations disponibles jusqu'alors. Ainsi, en conditionnant sur l'état précédent, on a :

$$P(X_t|O_{1:t-1}) = \sum_{x_{t-1}} P(X_t|x_{t-1}, O_{1:t-1}) P(x_{t-1}|O_{1:t-1})$$

   D'où l'on peut déduire l'équation de la prédiction en utilisant la propriété de Markov :

$$P(X_t|O_{1:t-1}) = \sum_{x_{t-1}} P(X_t|x_{t-1}) P(x_{t-1}|O_{1:t-1}) \tag{2.6}$$

2. L'étape de *mise à jour* exploite l'arrivée d'une nouvelle observation O_t et "corrige" la première estimation en utilisant généralement un *modèle d'observation*, permettant de calculer $P(O_t|X_t)$.

L'équation complète d'estimation du nouvel état au temps t devient alors la probabilité *a posteriori* suivante :

$$P(X_t|O_{1:t}) = \eta\, P(O_t|X_t) \sum_{x_{t-1}} P(X_t|x_{t-1}) P(x_{t-1}|O_{1:t-1}) \tag{2.7}$$

où η est une constante de normalisation, garantissant que la somme des probabilités partielles *a posteriori* sur les différents états est égale à 1. Ici :

$$\eta = \frac{1}{\sum_{x_t} P(O_t|x_t) P(x_t|O_{1:t-1})}$$

Difficultés d'un problème d'estimation classique

Le problème d'estimation repose donc sur :
- des *observations* et un *modèle d'observation* décrivant leur comportement vis-à-vis de l'état.
- un *modèle de dynamique* permettant d'effectuer une prédiction d'évolution de l'état sans la prise en compte d'une nouvelle observation, autrement dit un modèle de *transitions* entre les états.

Une première difficulté (ainsi qu'un point clef) dans un tel problème d'estimation est le *choix du modèle d'observation* : dans la pratique on a le plus souvent recours à une "mise en forme" préliminaire des données brutes pour qu'elles soient exploitables plus facilement et plus explicitement, mais il existe bien entendu de nombreuses manières (plus ou moins pertinentes) d'effectuer cette opération.
Une deuxième difficulté majeure, également point essentiel de la procédure d'estimation, est de définir un *modèle dynamique* approprié [Arnaud et al., 2005]. Celui-ci est le plus souvent défini *a priori*, à partir de connaissances initiales sur le type de dynamique ou d'une certaine expertise. Il est souvent aussi obtenu à partir d'un *apprentissage*. Or, nous avons vu qu'un tel apprentissage se révèle difficile à mettre en oeuvre, spécialement pour des applications de robotique d'extérieur, cadre de nos travaux.

2.3.2 Notions d'observation et de modèle dynamique dans notre application

Dans notre application nous manquons de connaissance *a priori* sur l'évolution des changements de modes, et donc sur la *dynamique*. Par contre, nous disposons d'une bonne connaissance des modes de déplacement utilisés, de leur fonctionnement et de leurs caractéristiques. Ceci nous permet en particulier de poser pour chacun des modes les bases d'un *modèle de fonctionnement nominal* et de générer des *signatures* pertinentes, afin d'évaluer le *comportement* du mode actif. Cette connaissance nous permet également de disposer d'une identification, pour chaque mode, de son contexte *dédié* (celui pour lequel il a été conçu) voire *nécessaire* (le mode ne peut pas s'appliquer dans un contexte différent).

Observations

L'application d'un mode de déplacement plutôt qu'un autre dépend entre autres choses du *contexte* dans lequel se trouve le robot, élément qui est généralement (au moins partiellement) observable. En effet par conception, les modes ont, pour certains, des *restrictions de contexte* dans leur bonne utilisation, ou plus simplement des *contextes dédiés*. Ainsi, un mode ayant des *restrictions de contexte* ne devra être utilisé que dans un contexte particulier (exemple : le terrain doit être plat). D'autre part, on préférera utiliser un mode dédié au contexte courant. Par exemple, si le terrain est plat, un mode dédié à ce contexte sera favorisé par le modèle d'observation par rapport à un mode de terrain accidenté car il génère moins de charge de calcul et permet de mettre en oeuvre des vitesses plus élevées (voir par exemple le mode *patinage* du Roller-Walker vs. le mode *marche*, section 1.2.4).

Notation : La variable d'observation du *contexte* à l'instant t sera notée O_t.

Modèle dynamique et transitions d'états

Les informations que nous exploitons pour "alimenter" les probabilités de transition sont les données sur le *comportement* du mode actif. Ces données ne sont bien sûr pas disponibles pour les autres modes, dans la mesure où l'on peut difficilement évaluer le comportement des modes qui ne sont pas actifs. Notre *modèle dynamique* est donc une composition d'un modèle posé *a priori* et des données de comportement, qui sont issues de la mise en forme d'observations obtenues en ligne.

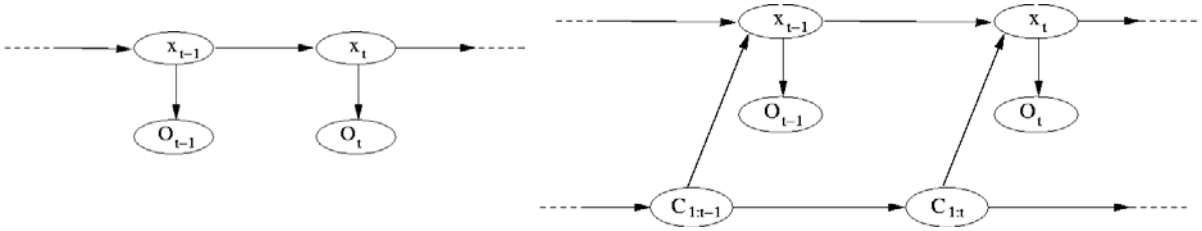
Intuitivement, si les données d'observation permettent d'adapter le mode à son contexte d'application, les données de comportement du mode courant vont influencer les transitions,

incitant le système à effectuer une transition vers un autre mode (qui soit également applicable dans le contexte actuel) si le mode courant révèle un mauvais comportement.

Notation : La variable de *comportement* à l'instant t sera notée par la suite C_t.

2.3.3 Estimation conditionnelle

Pour tenir compte des données de comportement dans l'estimation du mode à appliquer, données que nous considérons fondamentales pour choisir le bon mode et prendre en compte au mieux la réalité du terrain, nous posons notre problème sous une forme proche du *filtre conditionnel* de [Arnaud et al., 2005]. Nous introduisons une dépendance de l'état sur le *comportement* C_t. La propriété de Markov ainsi que l'indépendance conditionnelle sur les observations sont maintenues. La figure 2.6 propose le *graphe de dépendance orienté* de notre problème d'estimation conditionnelle, en comparaison avec celui d'un problème d'estimation "classique".



(a) Problème d'estimation "classique" (b) Notre estimation "conditionnelle"

FIG. 2.6 – *Graphe de dépendance orienté* de la chaîne de Markov cachée des variables $x_{0:t}$, $O_{1:t}$ (et $C_{1:t}$).

La distribution de probabilité *a posteriori* de l'état x_t à l'instant t étant données toutes les informations disponibles s'exprime $P(x_t|O_{1:t}, C_{1:t})$, à savoir la probabilité de l'état x_t sachant toutes les observations de contexte $O_{1:t}$ jusqu'au temps t et les données de comportement $C_{1:t}$ jusqu'au temps t. Si l'on connaît $P(x_{t-1}|O_{1:t-1}, C_{1:t-1})$, la probabilité de l'état à l'instant $(t-1)$, notre probabilité *a posteriori* de l'état est :

$$P(x_t|O_{1:t}, C_{1:t}) = \eta\, P(O_t|x_t, C_{1:t})\, P(x_t|O_{1:t-1}, C_{1:t})$$

Notons que η est une variable de normalisation qui n'a pas la même valeur que celle utilisée précédemment :

$$\eta = \frac{1}{\sum_{x_t} P(O_t|x_t, C_{1:t}) P(x_t|O_{1:t-1}, C_{1:t})}$$

Les deux étapes de l'estimation récursive bayesienne s'expriment alors ainsi :

1. **Prédiction** (prise en compte de la dynamique) :

$$P(x_t|O_{1:t-1}, C_{1:t}) = \sum_{x_{t-1}} P(x_t|x_{t-1}, C_{1:t}) P(x_{t-1}|O_{1:t-1}, C_{1:t-1})$$

   Nous voyons ici l'intervention des données de comportement dans les probabilités de changement de modes.

2. **Mise à jour** : après prise en compte de la nouvelle observation de contexte O_t, notre probabilité *a posteriori* devient :

$$P(x_t|O_{1:t}, C_{1:t}) = \eta\, P(O_t|x_t, C_{1:t})\, P(x_t|O_{1:t-1}, C_{1:t})$$

Or, les variables de comportement et d'observation sont *conditionnellement indépendantes* : en effet, $C_{1:t}$ ne transporte pas d'information sur O_t quand x_t est connu. On en déduit donc que $P(O_t|x_t, C_{1:t}) = P(O_t|x_t)$. Le modèle d'observation est donné par $P(O_t|x_t)$ à l'instant t.

Finalement, l'expression générale de la probabilité que le mode m_k soit celui à appliquer à l'instant t parmi les N possibles devient :

$$\forall k \in [\![1, N]\!],\ P(x_{k,t}|O_{1:t}, C_{1:t}) = \eta\, P(O_t|x_{k,t}) \sum_{i=1}^{N} P(x_{k,t}|x_{i,t-1}, C_{1:t}) P(x_{i,t-1}|O_{1:t-1}, C_{1:t-1}) \tag{2.8}$$

L'expression de la variable de normalisation η étant :

$$\eta = \frac{1}{\sum_{k=1}^{N} P(O_t|x_{k,t}) P(x_{k,t}|O_{1:t-1}, C_{1:t})} \tag{2.9}$$

Un exemple de HMM de la sélection de modes est illustré par la figure 2.7.

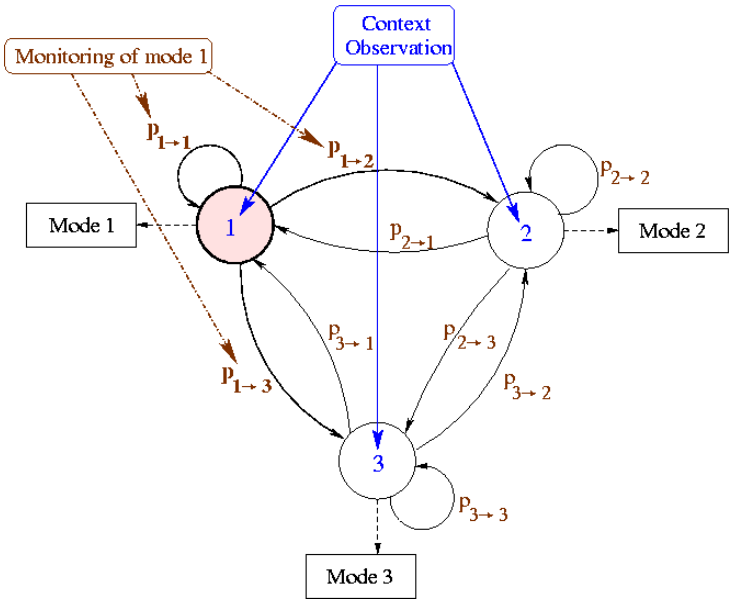


FIG. 2.7 – Un exemple de modèle de Markov caché pour l'estimation du mode à appliquer, avec trois modes disponibles et toutes les transitions possibles. Les données dites d'*observation* sont celles informant sur le *contexte* alors que le *monitoring* en ligne, fournissant les données de *comportement* du mode actif (ici le mode 1), renseigne sur les probabilités des transitions de l'état x_1 vers les autres états.

Pour pouvoir évaluer à chaque instant t toutes les probabilités partielles de modes à appliquer sachant les observations de contexte et les données de comportement, soit $P(x_{k,t}|O_{1:t}, C_{1:t})$ pour tout $k \in [\![1, N]\!]$, il nous faut donc calculer :

- les probabilités d'observation $P(O_t|x_{k,t})$, issues d'un modèle d'observation et d'informations de *contexte*,
- les probabilités de transitions conditionnelles $P(x_{k,t}|x_{i,t-1}, C_{1:t})$, pour tout $i \in [\![1, N]\!]$, à partir d'un modèle dynamique et de données de *comportement* du mode actif.

2.3.4 Observations : informations de contexte

La manière d'obtenir les données sur le terrain pour alimenter le modèle d'observation sera détaillée dans le chapitre 4. Deux types de données peuvent être potentiellement combinées, suivant leur disponibilité :

- des données *initiales* fournies par des opérateurs humains ou par exemple un autre robot tel qu'un robot aérien survolant la zone à parcourir (ou qui l'a survolée lors d'une mission de reconnaissance préalable),
- des données acquises en ligne par le banc de stéréo-vision et *pré-classifiées* en fonction des contextes et modes d'application.

2.3.5 Informations de comportement et transitions

Nous avons vu que la probabilité de transition (conditionnelle) d'un état x_i vers x_k s'exprimait $P(x_{k,t}|x_{i,t-1}, C_{1:t})$ (cf. équation 2.8), soit la probabilité de la transition *sachant le comportement* $C_{1:t}$.

En pratique ces probabilités de transition conditionnelles sont construites à partir des deux éléments suivants :

- un modèle dynamique fixé *a priori*, non conditionné sur les données de comportement,
- un *modèle de comportement* vis à vis des transitions, alimenté par des observations de comportement du mode courant.

Les données de comportement seront fournies par des *moniteurs* chargés d'évaluer le *comportement du mode courant* à partir des différences qui sont constatées avec le *comportement nominal* du mode, qui est supposé connu. Certains moniteurs sont spécifiques à un mode, étudiant quelques-unes de leurs caractéristiques propres. D'autres peuvent être plus génériques et s'appliquer à plusieurs modes ayant des caractéristiques communes.

Nous ne disposons pas de données de comportement pour les modes qui ne sont pas actifs. Par conséquent, si m_c est le mode actif à l'instant $(t-1)$,

$$\forall\, i \neq c, \quad P(x_{k,t}|x_{i,t-1}, C_{1:t}) = P(x_{k,t}|x_{i,t-1})$$

Un modèle dynamique *a priori*, non conditionné sur les données de comportement, est posé empiriquement. Il fournit l'ensemble des probabilités $P(x_t|x_{t-1})$, spécifiant en particulier les probabilités nulles pour les transitions non réalisables ou interdites. Pour tout x_{t-1} ne concernant pas un mode actif, ces probabilités sont exactement les probabilités de transitions intervenant dans le HMM de sélection de mode.
Ce modèle dynamique *a priori* est défini de manière à éviter des changements de mode trop rapides ou non nécessaires (filtrage).

Modèle de comportement

Les données de comportement viennent s'*ajouter* à ce modèle dynamique a priori lorsqu'elles sont disponibles. Leur rôle est d'*accentuer* la probabilité de transition vers d'autres modes si

un mauvais comportement est constaté, d'autant plus que le comportement réel est éloigné du comportement nominal attendu.

La variable aléatoire de comportement $C_{1:t}$ traduit précisément cet éloignement. Ainsi, lorsque le comportement semble être parfaitement celui attendu, la probabilité de mauvais comportement est nulle. On a alors, pour tout k tel que le mode m_k est un successeur possible du mode courant m_c :

$$P(x_{k,t}|x_{c,t-1}, C_{1:t}) = P(x_{k,t}|x_{c,t-1})$$

Si au contraire le comportement n'est pas *nominal*, la probabilité de transition conditionnelle de x_c vers x_k, sachant que le mode courant est m_c, s'exprimera :

$$\forall k \neq c, \quad P(x_{k,t}|x_{c,t-1}, C_{1:t}) = P(x_{k,t}|x_{c,t-1}) + (1 - P(x_{k,t}|x_{c,t-1}))Q_{c,k}(C_{1:t}) \tag{2.10}$$

où $Q_{c,k}(C_{1:t})$ est une pseudo-probabilité comprise entre 0 et 1. Ainsi, avec cette formulation, la probabilité de transition conditionnelle est telle que :

$$P(x_{k,t}|x_{c,t-1}) \leq P(x_{k,t}|x_{c,t-1}, C_{1:t}) \leq 1$$

La grandeur $Q_{c,k}(C_{1:t})$ dépend du comportement du mode m_c et de la "stratégie" de changement de mode (quels sont les modes disponibles en cas d'échec de m_c et quel type de comportement doit favoriser quel mode) : cette grandeur est fournie par les moniteurs du mode courant.
Par exemple, dans le cas où seul un mode de déplacement m_k est disponible pour suppléer le mode m_c en difficulté (il existe un seul $k \neq c$ tel que la transition $(x_k|x_c)$ est réalisable, voir figure 2.8), $Q_{c,k}(C_{1:t})$ traduira la probabilité de comportement de m_c. Si ce comportement tend vers la faute, la probabilité de transition $P(x_{k,t}|x_{c,t-1}, C_{1:t})$ tendra vers 1 (équation 2.10), favorisant fortement un changement de mode en faveur de m_k.
Dans le cas où plusieurs modes m_k sont des successeurs possibles du mode courant m_c, une stratégie devra être définie pour déterminer les transitions à favoriser en fonction des différents comportements révélés par les moniteurs.

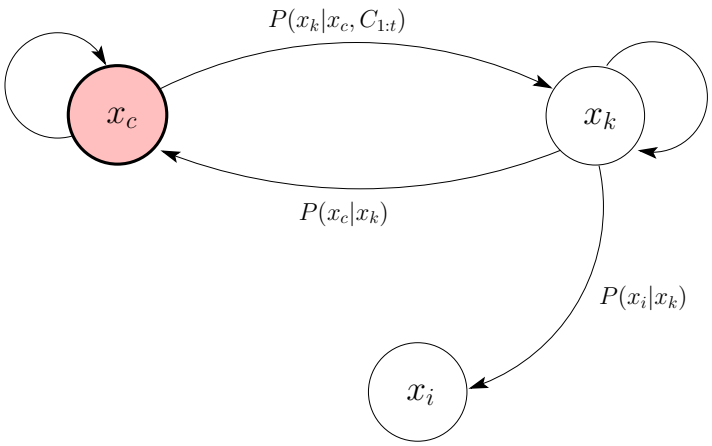


FIG. 2.8 – Exemple simple de transition conditionnelle : le mode actif est m_c (on dispose donc de données sur son comportement) et le seul mode "alternatif" est m_k.

Nous verrons en détails dans les chapitres 5 et 6 comment sont générées les données de comportement en pratique et comment sont définies ces stratégies de transition suivant les modes et les moniteurs disponibles.

Transitions

Le modèle dynamique devra faire en sorte de ne pas générer de trop fréquents changements de modes si cela n'est pas nécessaire. En effet, des changements de modes peuvent parfois nécessiter des opérations plus ou moins coûteuses telles que des ré-initialisations de fonctionnalités. Ils peuvent même, le cas échéant, nécessiter un *arrêt complet du robot*. Par exemple, dans le cas de lois de commande différentes pour la locomotion, garantir la stabilité et la continuité lors d'une transition dynamique d'une commande à une autre est un problème délicat.

Il convient également de s'assurer d'éviter des comportements *instables* avec succession de changements d'un mode à un autre.

Des ajustements de ce modèle dynamique *a priori* et de la prise en compte des données de comportement peuvent permettre d'agir sur le type global de comportement souhaité pour le robot. Ainsi peut-on mettre l'accent sur la rapidité ou la sécurité, sur la conservation des modes ou sur le changement à la moindre difficulté...

Bilan

Les probabilités de transitions sont donc initialement issues d'un modèle dynamique *a priori*, auquel nous ajoutons une influence du comportement du mode courant selon le principe suivant, en notant m_c le mode actif. Dans la chaîne de Markov, les probabilités de transition associées à tous les arcs sortant d'états x_k qui concernent des modes non actifs sont celles issues du modèle dynamique *a priori* : $P(x_{k,t}|x_{i,t-1})$ avec $i \neq c$. Les probabilités de transition associées quant à elles aux arcs sortant de l'état x_c sont conditionnées sur le comportement de m_c, qui influence ces transitions en augmentant leur probabilité lorsque le comportement n'est pas satisfaisant.

2.4 Conclusion

Nous avons présenté dans ce chapitre l'approche que nous proposons pour réaliser une estimation du *mode de déplacement* à appliquer sur le robot, de manière réactive. Cette estimation repose sur :

- des informations de *contexte*, à savoir le type de terrain sur lequel le robot est en train d'évoluer,
- des informations de *comportement* du mode actif, combinées à un modèle dynamique *a priori*.

Afin de mettre en oeuvre ce système, il nous faut donc les éléments suivants :

- des **modes de déplacement**, avec une bonne connaissance de leurs caractéristiques, de leur contexte adapté ou dédié et de leur fonctionnement pour pouvoir identifier un modèle de *fonctionnement nominal*. Les modes de déplacement que nous avons exploités et mis en oeuvre, qui seront distingués en modes de navigation et modes de locomotion, sont présentés dans le chapitre 3 ;
- des moyens d'observation du **contexte terrain** afin d'alimenter le *modèle d'observation* de notre processus d'estimation. Cela fera l'objet du chapitre 4 ;
- un modèle dynamique *a priori* et des **moniteurs** pour effectuer une **évaluation du comportement** de chacun des modes disponibles lorsqu'il est actif et pour mettre à jour en ligne les probabilités de transition entre les modes. Le développement de tels moniteurs fait l'objet du chapitre 5.

Des résultats de la sélection de mode complète faisant intervenir tous ces éléments seront proposés dans le chapitre 6.

Chapitre 3

Modes de déplacement

Comme nous l'avons vu dans le chapitre précédent, le principal objectif de ces travaux est de mettre en oeuvre un système de sélection de *mode de déplacement* d'un robot d'extérieur reposant sur :

- une estimation du *contexte* dans lequel le robot évolue,
- une évaluation du *comportement* du robot.

Il nous faut donc avant tout des plates-formes robotiques disposant de plusieurs modes de déplacement (navigation et/ou locomotion). Une connaissance suffisamment précise de ces plates-formes et du fonctionnement de ces modes est nécessaire afin d'identifier :

- le contexte dans lequel un mode donné devra être utilisé,
- le *comportement nominal* de ce mode, référence à partir de laquelle nous pourrons évaluer la qualité de son comportement effectif.

Ce chapitre vise donc à présenter les plates-formes expérimentales utilisées, puis les modes de navigation et de locomotion dont elles disposent.

3.1 Présentation des plates-formes

Nous présentons tout d'abord dans cette section les plates-formes exploitées dans ces travaux. Les travaux expérimentaux concernent essentiellement les deux robots terrestres du LAAS-CNRS : Dala et Lama. Le premier sera surtout exploité pour la partie *navigation*, le deuxième plutôt pour la partie *locomotion*. Le robot Hylos 2, développé par nos collègues du Laboratoire de Robotique de Paris, a également inspiré une part de ces travaux. Toutefois, sa "naissance" étant encore bien récente, les développements le concernant ne sont encore que préliminaires. Il sera donc présenté plus sommairement, avec les modes de locomotion qui lui sont propres, en annexe B.

3.1.1 Dala

Le robot Dala (figure 3.1) est un *ATRV*[1] développé par la société *iRobot*, arrivé au LAAS début 2002. Ce type de robot est composé d'un châssis rigide, non articulé et sans suspension, avec quatre roues équipées de pneus sculptés. Il s'agit donc d'un robot d'extérieur mais qui n'est pas destiné aux terrains vraiment très accidentés.

[1] *All Terrain Robot Vehicle*

FIG. 3.1 – Le robot Dala

Capteurs

Dala a été équipé par le LAAS des capteurs suivants :

- une **centrale inertielle** trois axes (communément désignée par le sigle IMU[2]) fournissant une mesure des accélérations suivant les trois axes liés au robot et des vitesses angulaires autour de ces mêmes axes,
- des **codeurs optiques** permettant de réaliser une mesure de la distance parcourue par les roues,
- un **gyromètre** optique précis et de faible dérive, effectuant une mesure de la vitesse de rotation du robot autour de son axe vertical,
- un **banc de stéréo-vision** constitué de deux caméras numériques CCD couleur de résolution maximale 1024x768,
- un **capteur laser** $SICK$, situé à l'avant du robot. Il effectue des balayages à 180 degrés dans un plan avec une résolution de 10 mm (donnée constructeur) à une fréquence de 10 Hz,
- une ceinture (partielle) de **capteurs ultra-sons**, non exploités dans ces travaux.

Le tableau suivant résume les données et capteurs *proprioceptifs* disponibles sur Dala :

[2]Inertial Measurement Unit

Donnée	Notation	Capteur
Accélération en x	a_x	Accéléromètre (IMU)
Accélération en y	a_y	Accéléromètre (IMU)
Accélération en z	a_z	Accéléromètre (IMU)
Vitesse angulaire autour de x	ω_x	Gyromètre (IMU)
Vitesse angulaire autour de y	ω_y	Gyromètre (IMU)
Vitesse angulaire autour de z	ω_z	Gyromètre (IMU)
Position angulaire des moteurs		Codeurs optiques

Locomotion

La vitesse maximale de Dala est de $2\,m/s$ (soit $7,2\,km/h$), ce qui constitue une vitesse relativement élevée pour un robot autonome évoluant en environnement naturel quelconque. C'est un robot de type *skid-steering* (de type "char") : ses roues sont non directionnelles, il utilise pour tourner un différentiel de vitesses sur les roues de chaque côté (voir 3.3.1).

Notons qu'au cours de cette thèse des développements ont également été réalisés sur le robot Gromit du *NASA Ames Research Center* en Californie. Gromit est issu de la même famille que Dala : il s'agit d'un *ATRV Junior*, qui a donc la même structure mécanique que Dala avec des dimensions inférieures. Il n'est pas équipé de capteur laser, mais dispose par ailleurs de capteurs équivalents à Dala, auxquels s'ajoute un compas magnétique 3 axes fournissant des mesures d'orientations absolues.

3.1.2 Lama

Le robot LAMA[3] (figure 3.2) a été équipé par Alcatel-Espace et par le LAAS à partir de fin 1996 pour réaliser un démonstrateur pour des missions d'études sur Mars et dans des milieux hostiles. Après une belle carrière au service des expérimentations du projet EDEN au laboratoire, conclue par le développement d'un moniteur de locomotion (section 5.2), il coule désormais une retraite "dorée" au Musée des Arts et Métiers à Paris depuis la fin 2005.

Châssis

Lama repose sur une structure "tout–terrain" de conception Russe appelée *Marsokhod*, composée de trois essieux et d'un châssis poly-articulé. Chaque essieu comporte deux roues cylindro-coniques constituées de feuilles d'alliage de titane. Chaque roue contient un jeu de batteries et un moteur à courant continu. Le châssis possède deux articulations motorisées qui permettent au robot de se déplacer en mode *péristaltique* dans lequel le robot "rampe" comme une chenille (3.3.2). Les autres articulations sont passives.

Capteurs

Lama dispose des capteurs suivants :

- des **capteurs de configuration** qui permettent de connaître les trois angles qui définissent la configuration du châssis (figure 3.2). Il s'agit de potentiomètres fournissant une mesure d'angle avec une résolution de l'ordre du 1/100 de degré. Ces trois angles correspondent aux articulations *passives* du châssis. Deux autres potentiomètres permettent de connaître

[3] à l'origine, acronyme pour *Lavochkin Alcatel Model Autonomous*

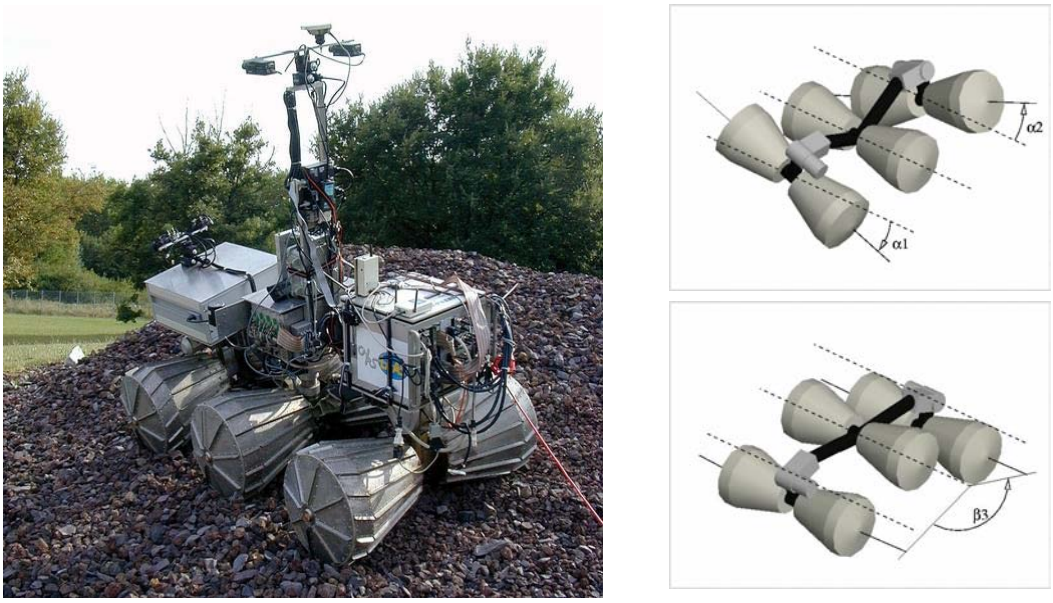


FIG. 3.2 – Le robot Lama et son châssis articulé

les angles formés par les bras du robot, qui sont activement contrôlés lors de l'application du mode de locomotion *péristaltique*.

- **des codeurs odométriques** incrémentaux qui délivrent 500 impulsions par tour. Ils sont montés sur l'arbre rapide du moteur (avant réduction) : le rapport des réducteurs étant de 1/503, ils délivrent 251 500 impulsions par tour de roue, ils sont donc extrêmement précis.
- un **gyromètre** à fibre optique (modèle KVH E-Core 1000), qui fournit une estimée de la vitesse angulaire du robot. Les données sont produites à $10\,Hz$ avec une résolution légèrement supérieure à $0,001\ deg.s^{-1}$. Les erreurs sur les valeurs mesurées suivent une gaussienne dont l'écart-type est de l'ordre de $0,015\ deg.s^{-1}$. Nous disposons donc d'une mesure de la vitesse de rotation du robot bénéficiant d'une très bonne précision.
- **un système de localisation GPS** différentiel à mesure de phase (modèle Leica MC1000), qui fournit une estimée de position à $10\,Hz$, dont la précision est de $1\,cm$ à $\pm 3\,\sigma$ sur les valeurs X et Y, et de $2\,cm$ à $\pm 3\,\sigma$ sur Z (altitude).
- deux **bancs de stéréo-vision :** montés chacun sur une tourelle orientable suivant deux axes concourants. En pratique, seul celui monté sur le mât central a été exploité dans nos applications.

Le tableau suivant résume les données et capteurs *proprioceptifs* disponibles sur Lama :

Donnée	Notation	Capteur
Site essieu central	ϕ	Inclinomètre [-30°,+30°]
Gîte essieu central	ψ	Inclinomètre [-30°,+30°]
Vitesse angulaire essieu central	ω_{Gyro}	Gyromètre
Angle roulis essieu avant	α_1	Potentiomètre
Angle roulis essieu arrière	α_2	Potentiomètre
Angle articulation avant	β_1	Potentiomètre
Angle articulation arrière	β_2	Potentiomètre
Angle articulation centrale	β_3	Potentiomètre
Position angulaire des moteurs		Codeurs optiques

Locomotion

La vitesse maximale de Lama est seulement de $0,17\,m/s$, il est donc plutôt destiné à des terrains vraiment difficiles. Il s'agit également d'un robot de type *skid-steering* (de type "char") car ses roues sont non directionnelles. Il est donc commandé en vitesse de la même manière que son compère Dala.

3.2 Les modes de navigation

Nous présentons les modes de déplacement qui sont exploités dans ces travaux et dont disposent les plates-formes présentées ci-dessus. Ces méthodes ont été développées par des travaux antérieurs au laboratoire et adaptées aux différentes plates-formes et à nos besoins. Nous verrons d'abord dans cette section les méthodes dites de *navigation* (perception et décision sur le mouvement) puis dans la section suivante des modes de *locomotion* (réalisation du mouvement). Dans notre cas, le rôle pratique d'un mode de *navigation* est de fournir régulièrement des consignes de vitesses au robot sous la forme du couple vitesse linéaire/angulaire (v, ω) qui permettront à terme d'atteindre le but (x, y) donné par un opérateur ou une couche décisionnelle (planificateur de tâches par exemple), tout en évitant les obstacles présents dans l'environnement.

3.2.1 Navigation en terrain plat : *FlatNav*

Ce mode de navigation est utilisable sur des robots équipés d'un capteur laser 2D, donc dans notre cas il est disponible *uniquement sur le robot Dala*. Il s'agit d'un mode de mouvement réactif exploitant les informations sur les obstacles environnants fournies par le capteur laser 2D, adapté de [Minguez et al., 2004]. Du fait que ce capteur, positionné à l'horizontale sur le robot, ne peut repérer que les obstacles situés dans un plan, ce mode ne doit être appliqué que dans un contexte de terrain plat, d'où son nom de *FlatNav*, pour Flat Terrain Navigation Mode en anglais (par opposition au *RoughNav* présenté dans la section suivante). Le principe de fonctionnement général de ce mode est illustré par la figure 3.3.

Perception : laser 2D et interprétation des données

Le capteur laser (situé à l'avant du robot) effectue un balayage périodique à 180 degrés et donne ainsi une mesure de la distance et de la position relative des obstacles détectés dans le plan du capteur (fig. 3.4). Ces données sont intégrées par un module fonctionnel nommé `aspect` pour construire une carte locale de l'environnement dans un voisinage proche du robot, mise à jour à une fréquence relativement élevée, permettant ainsi de prendre en compte des obstacles

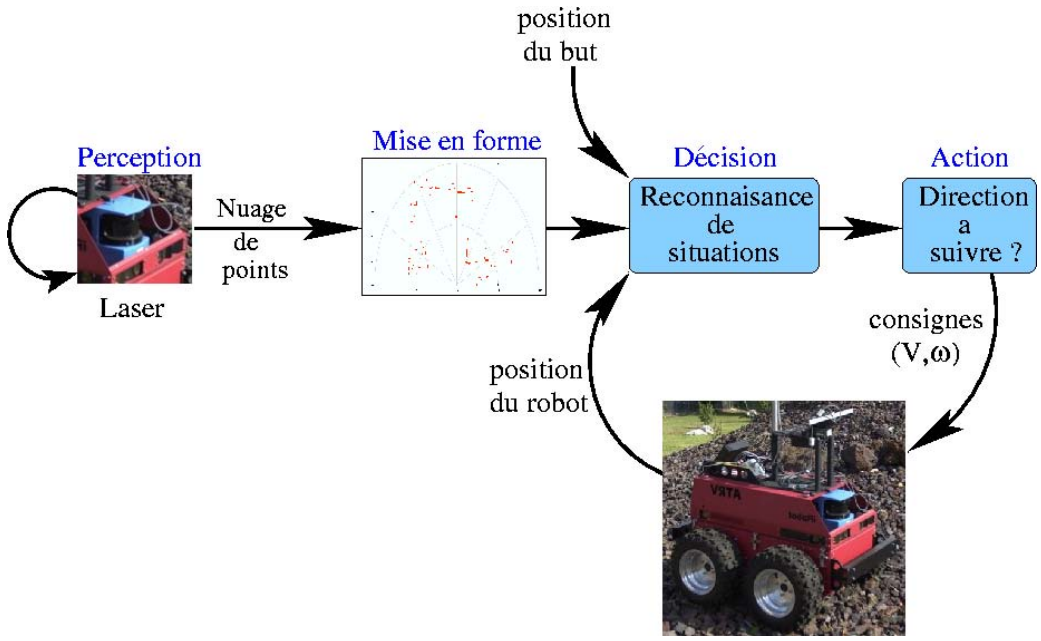


FIG. 3.3 – Boucle de navigation perception/décision/action dans le mode *FlatNav*

statiques comme dynamiques. Dans cette carte, les nuages de points repérés par le laser sont filtrés et mis en forme pour faire apparaître plus nettement les obstacles (après regroupements de points) et les espaces libres.

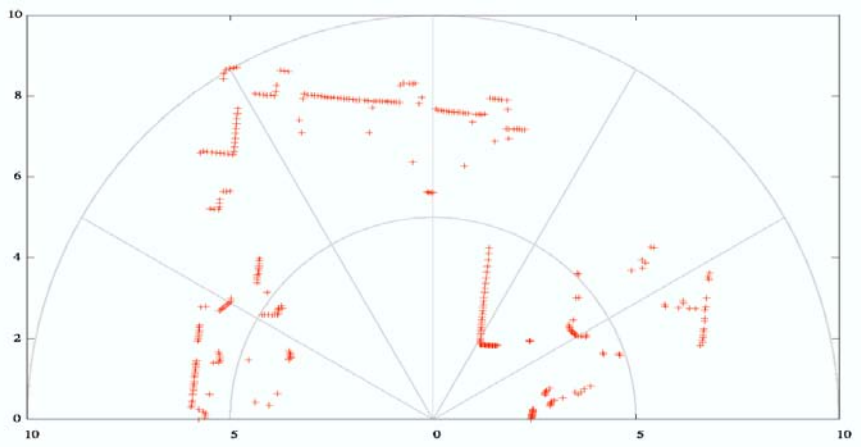


FIG. 3.4 – Exemple de données brutes fournies par un balayage laser

Mouvement

La méthode de mouvement réactif tirant profit de ces informations est une évolution de la méthode *Nearest Diagram* (ND) [Minguez et al., 2001] nommée *ND+* ou *NDD*. Afin de pallier les habituels défauts des méthodes locales d'évitement d'obstacle, elle propose une stratégie de type "divide and conquer" (diviser pour mieux régner), reposant sur l'identification de catégories prédéfinies de situations de navigation et l'application d'une heuristique de mouvement corres-

pondante [Minguez et al., 2004]. La méthode se décompose en trois phases :

Phase 1 : Mise en forme des données La première étape de la méthode consiste en l'exploitation et l'interprétation des données sur les obstacles fournies par le laser. En premier lieu une *zone de sécurité* est définie autour du robot. On recherche ensuite les *discontinuités* entre les obstacles et on définit des *régions* entre deux discontinuités contigües (voir figure 3.5), repérant particulièrement celles qui sont *traversables* par le robot. Enfin, la région ainsi "navigable" qui est la plus proche du but est sélectionnée. C'est la *région de déplacement libre* (*free walking area*, telle qu'elle est définie dans [Minguez et al., 2004]).

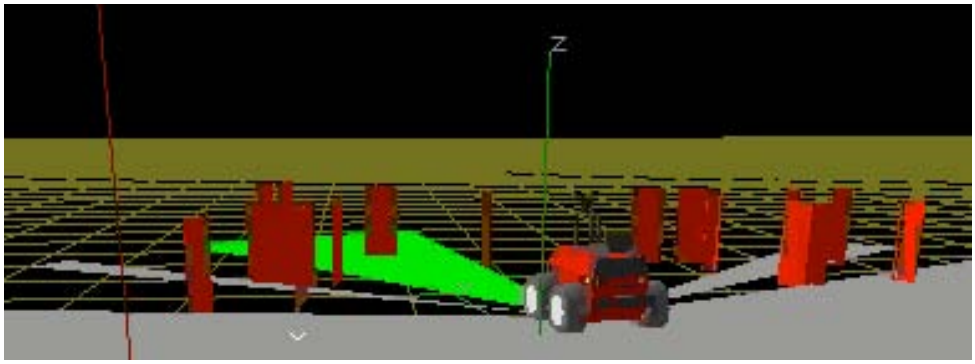


FIG. 3.5 – Illustration des données du laser mises en forme pour NDD : on constate ici quatre régions extraites

Phase 2 : Sélection de la situation L'ensemble des situations possibles est divisé en une *partition* (représentation *complète* et *exclusive*). Ces différentes situations sont définies à partir de *critères* spécifiques :

1. Critère de *sécurité* : présence ou non d'obstacles dans la zone de sécurité du robot,
2. But situé ou non dans la *Région de Déplacement Libre* (RDL),
3. Critère de *largeur de la Région de Déplacement Libre* : cette dernière est considérée comme *large* ou *étroite* suivant que sa largeur est supérieure ou inférieure à un seuil prédéfini,
4. Critère de *distribution des obstacles* : cette distribution est considérée comme *dangereuse* s'il y a des obstacles dans la zone de sécurité des deux côtés de la discontinuité (la plus proche du but) de la région de déplacement libre, et au contraire plus sûre si les obstacles ne sont que d'un côté de la discontinuité.

On définit ainsi, en utilisant ces critères, une représentation complète et exclusive des situations par un ensemble de six situations, nommées comme suit :

- HSGR : *High Safety Goal in Region* (Haute sécurité et but dans la RDL),
- HSWR : *High Safety Wide Region* (Haute sécurité et RDL large),
- HSNR : *High Safety Narrow Region* (Haute sécurité et RDL étroite),
- LSGR : *Low Safety Goal in Region* (Faible sécurité et but dans la RDL),
- LS1 : *Low Safety 1 side* (Faible sécurité 1 côté),
- LS2 : *Low Safety 2 sides* (Faible sécurité 2 côtés).

La figure 3.6 présente la manière dont sont utilisés les critères précisés plus haut pour différencier ces situations.

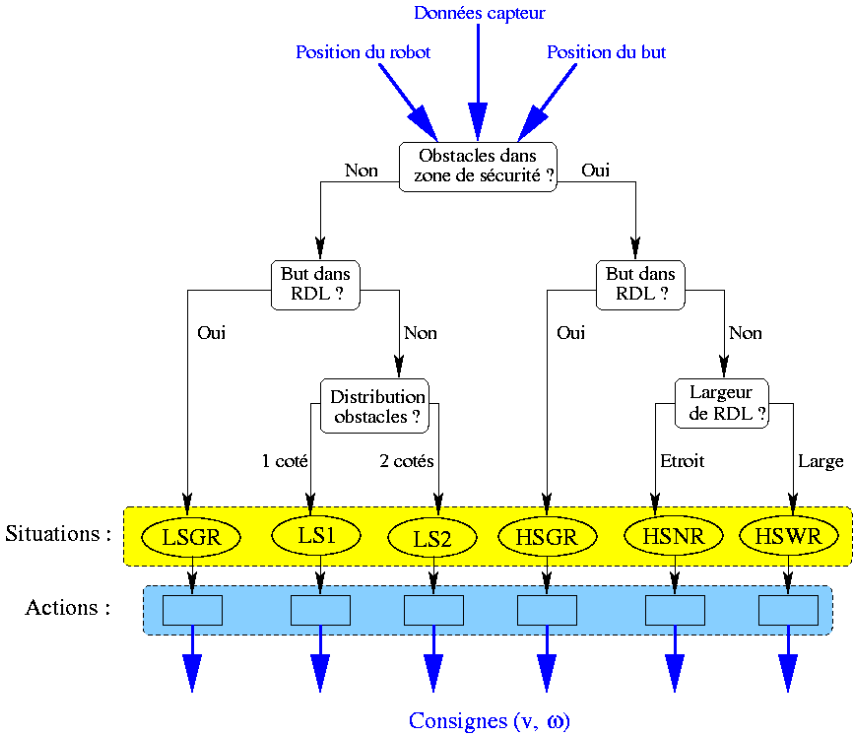


FIG. 3.6 – Sélection des *situations* pour NDD. Nous notons RDL la *Région de Déplacement Libre*. A chaque situation est associée une action spécifique correspondant à une manière de générer les consignes de vitesses pour le robot.

Phase 3 : Actions pour le mouvement Une action spécifique est associée à chacune de ces situations. Elle consiste à calculer la direction θ_{robot} suivant laquelle le robot va devoir se diriger. Dans le cas d'une situation simple comme HSGR cette direction sera définie comme celle du but. Dans des cas où l'environnement proche du robot est plus encombré par des obstacles, cette direction est corrigée suivant la proximité des obstacles, leur situation, ou encore la largeur de la *Région de Déplacement Libre*. θ_{robot} étant déterminé, cette direction est alors transcrite en des consignes de vitesses (v, ω) en tenant compte dans notre cas de la forme et de la non-holonomie du robot (Dala) ainsi que des vitesses maximales (v_{max}, ω_{max}) que nous lui imposons. Ces consignes sont alors envoyées au module de locomotion, chargé de l'exécution de ce mouvement.

Remarques sur ce mode de déplacement

Avantages de la méthode Parmi les points positifs de cette méthode vis-à-vis des méthodes classiques de mouvement réactif, on peut citer [Minguez et al., 2004] :

- les situations de "pièges locaux", que constituent généralement les obstacles en forme de U, sont bien gérées,
- le mouvement apparent est dépourvu d'oscillations et autres instabilités,
- la sélection des directions très éloignées de celle du but, quand cela peut être utile, n'est pas un problème (le robot peut par exemple se déplacer un temps en direction d'un obstacle si celui-ci est suffisamment éloigné),
- il y a très peu de paramètres internes à ajuster,

- les actions sont continues entre les transitions "courantes" (telles que $HSGR \leftrightarrow LSGR$, $HSWR \leftrightarrow LS1$ ou $HSNR \leftrightarrow LS2$),
- La méthode est simple et rapide en calcul, elle permet donc de faire évoluer le robot avec des vitesses maximales relativement élevées si l'environnement et la répartition des obstacles s'y prêtent.

Notons que cette méthode de navigation pourrait également être mise en oeuvre en utilisant d'autres types de données que celles fournies par un capteur laser. Il suffirait alors de les mettre en forme d'une manière similaire.

Limitations de la navigation locale Bien que cette méthode semble bien se comporter face à des obstacles "en forme de U", elle reste une méthode *locale*, incompatible avec des objectifs de navigation plus *globale*. Les buts qu'elle reçoit doivent donc rester des buts *locaux*, fournis le cas échéant par une méthode plus globale chargée de générer une série de sous-buts permettant d'amener le robot au but final.

"Multi-modalité" Nous pouvons remarquer que ce mode de navigation utilise lui-même des "modes" différents, correspondant en réalité à des actions distinctes (concernant la décision pour le mouvement à réaliser). La partie perception est, elle, toujours la même. Le choix du "mode" est effectué directement à partir d'une simple reconnaissance de *situations*, donc par la perception du *contexte* (sur la base d'informations extéroceptives sur l'environnement), par une méthode purement *déterministe*. Les incertitudes sur l'estimation du contexte ne sont pas prises en compte et le choix de l'action n'est en aucun cas dépendant du *comportement* du robot.

3.2.2 Navigation en terrain accidenté : *RoughNav*

Ce mode, exploitable sur les robots Dala et Lama, a été conçu pour naviguer dans des terrains accidentés (d'où son nom de *RoughNav* pour *Rough Terrain Navigation Mode*). Il exploite la stéréo-vision, une carte d'élévation et une méthode de planification locale.
Initialement développé pour des robots Marsokhod tels que Lama, nous l'avons adapté (pour son utilisation sur des robots type *ATRV* comme Dala) et complété pour les besoins de ces travaux de thèse.

Perception : Modèle Numérique de Terrain

Le mode *RoughNav* repose avant tout sur une manière de percevoir et de représenter l'environnement. Le capteur utilisé est un banc de stéréo-vision qui permet d'obtenir des données *denses* en trois dimensions.
Lorsqu'une paire d'images est acquise par les deux caméras du banc, les algorithmes de stéréo-vision effectuent un *appariement* des pixels communs à ces deux images. Suivant leur position respective dans les deux images, on peut alors associer à chacun de ces pixels une *disparité* (figure 3.8) qui correspond à une différence de positionnement dans les deux images. Si la transformation géométrique entre les deux caméras du banc de stéréo-vision est connue, à chaque pixel ainsi *corrélé* peut être attribuée une profondeur. Finalement, connaissant les paramètres intrinsèques et extrinsèques de calibration, on peut associer à chaque pixel des coordonnées 3D complètes [Lemonde, 2005].
Le terrain est ici représenté sous la forme d'une *carte d'élévation*, ou *Modèle Numérique de Terrain* (MNT). Le terrain est découpé en une grille cartésienne régulière (x, y) définie dans un plan horizontal, similaire à ce qui est utilisé dans le cas des grilles d'occupation. Une acquisition

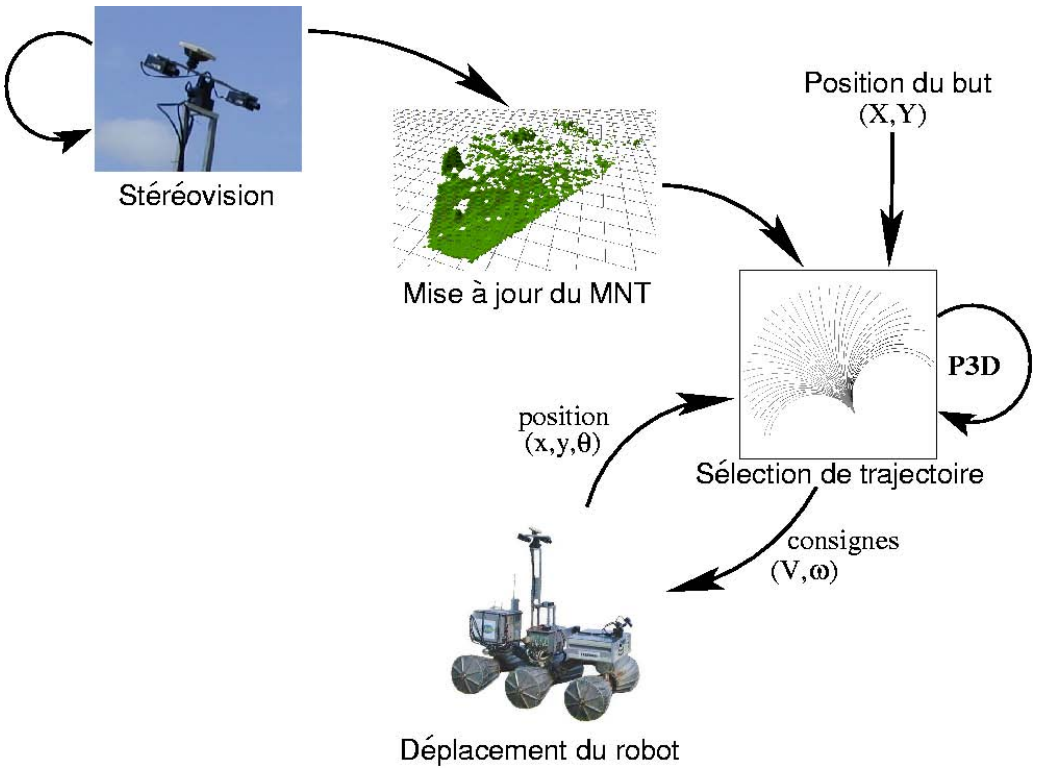


FIG. 3.7 – Boucle de navigation perception/décision/action dans le mode *RoughNav*

du banc de stéréo-vision génère donc un nuage de points 3D qui sont positionnés dans cette grille cartésienne. On affecte à chaque cellule (x_i, y_j) l'altitude moyenne z_m de tous les points positionnés dans cette cellule ainsi que son écart-type σ_z. Lors d'une nouvelle acquisition, les nouveaux points identifiés comme appartenant à chaque cellule sont fusionnés avec les précédents en moyennant les altitudes. Chaque cellule du MNT contient donc les informations suivantes :

- le nombre de points 3D qui ont été inclus (fusionnés) dans cette cellule,
- l'**élévation moyenne** de ces points : z_m,
- l'**écart-type** sur cette élévation σ_z,
- une confiance sur les informations d'élévation.

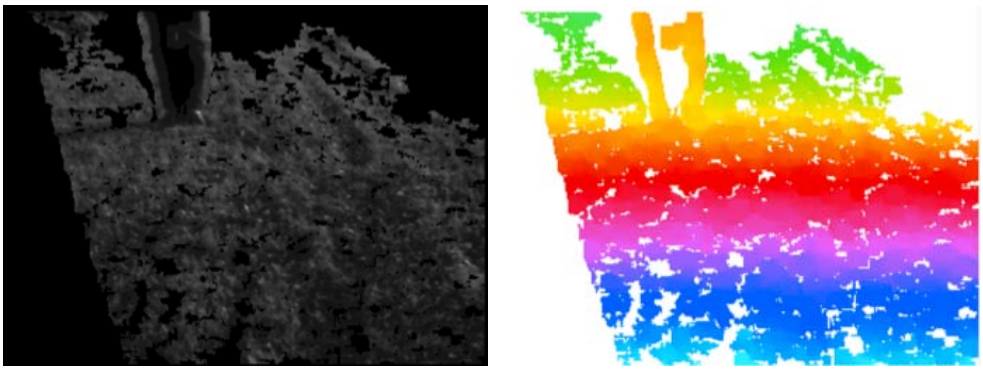
La figure 3.9 présente un exemple de MNT obtenu après plusieurs acquisitions depuis *une* position du robot.

Génération de mouvement : planification locale par *P3D*

L'étape de génération de mouvement (ou sélection de trajectoire locale) consiste en l'évaluation d'un *risque* et d'un *intérêt* le long d'un ensemble de trajectoires élémentaires, sous la forme d'arcs générés à partir de la position du robot (figure 3.10) [Bonnafous et al., 2001]. Le *risque* est lié à la difficulté du terrain à traverser et l'*intérêt* dépend du rapprochement du but. La trajectoire sélectionnée est celle qui optimise un critère *intérêt/coût*. Cette méthode est réalisée par un module fonctionnel appelé *P3D*.

L'évaluation des arcs se fait de la manière suivante : le long de chaque arc une série de noeuds est répartie régulièrement, correspondant à des positions sur le MNT. En chacune de ces positions, une fonction de placement de la structure 3D du robot sur le MNT est appliquée pour évaluer

(a) Image gauche rectifiée (b) Image droite rectifiée



(c) Pixels appariés (d) Image de disparités

FIG. 3.8 – Obtention d'une image de *disparité* à partir de l'*appariement* des pixels présents dans la paire d'images acquises par un banc de stéréo-vision.

la configuration et l'attitude qu'aurait le robot s'il était positionné à cet endroit du MNT. Un *risque* lié est alors calculé, ainsi qu'un risque dû à l'enchaînement des configurations et attitudes successives. Le risque associé à l'arc est le résultat de l'intégration de ces risques élémentaires le long de cet arc. Nous détaillons plus avant les étapes de ce processus, en particulier l'opération de *placement* du robot sur le MNT, vu l'importance qu'elle aura par la suite.

Placement du robot et prédiction de l'attitude/configuration La fonction de placement du robot (`placeRobot`) estime une configuration et une attitude prédites pour le robot en une position donnée (x, y, θ) sur le MNT (fig. 3.12). Elle est bien entendu fortement dépendante de la structure du robot et en particulier de la géométrie de son châssis. Même si elle nécessite l'adaptation de la fonction à la structure du robot considéré, restreignant ainsi la généralité, cette démarche nous paraît judicieuse car elle permet de rester au plus proche de la réalité du terrain, rendant la prédiction d'autant plus pertinente. Cette fonction a été initialement développée pour le robot Lama. Nous l'avons ensuite adaptée aux structures du type du robot Dala dans le cadre de ces travaux de thèse.
Le placement du robot repose tout d'abord sur une fonction de placement d'un essieu du robot (`placeAxle`), elle-même faisant appel itérativement à une fonction de placement des roues (`placeWheel`). Estimer les angles de configuration et d'attitude complets consiste à placer correctement sur le MNT :

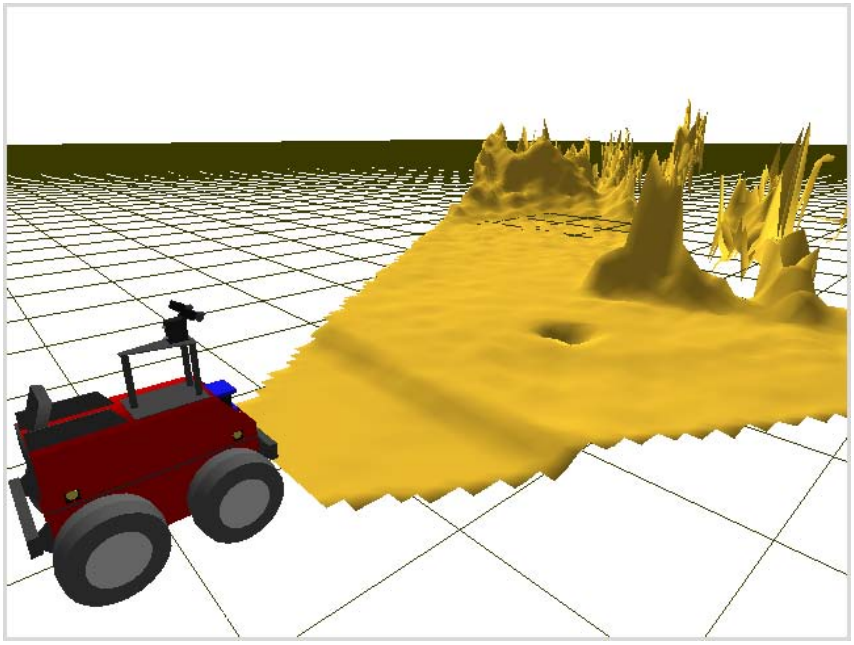


FIG. 3.9 – Un exemple de MNT réalisé par Dala depuis *une* position.

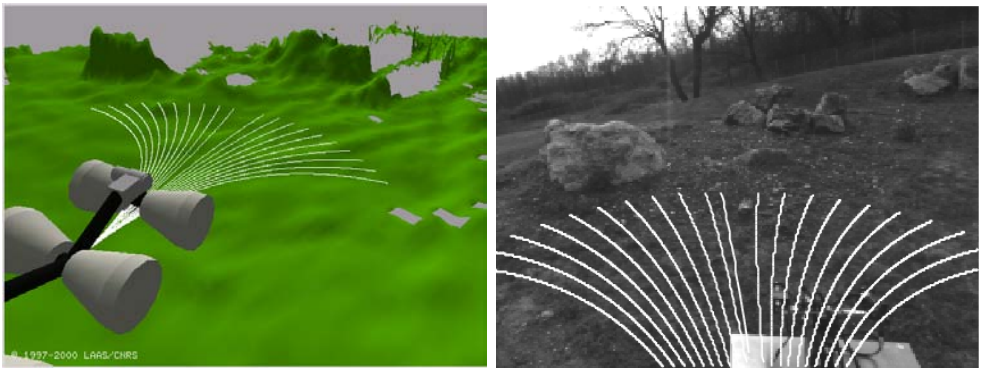


FIG. 3.10 – Ensemble d'arcs évalués devant le robot par *P3D*

– pour Lama : les trois essieux avant-milieu-arrière (avec contraintes entre eux),
– pour Dala : les deux essieux contraints (définissant ainsi le positionnement de la caisse).

Le principe de la fonction `placeAxle` est le suivant (figure 3.11) :
On procède tout d'abord au placement des deux roues de l'essieu sur le relief du MNT, avec une orientation initiale horizontale. Ce placement donne l'élévation de chacune des roues ainsi placées. La distance d entre les axes des deux roues permet de définir une nouvelle orientation d'essieu. On effectue alors un nouveau placement des deux roues suivant cette orientation, réitérant le processus jusqu'à la stabilisation à $d < \epsilon$, où ϵ est un seuil de tolérance défini initialement. La fonction de placement d'une roue (`placeWheel`) est donc susceptible d'être appelée à de nombreuses reprises lors du placement complet du robot. Elle exploite une discrétisation des élévations du profil de la roue, représentée par un tableau rectangulaire (le profil d'une roue de Lama est de type conique, celui d'une roue de Dala est plus cylindrique), avec une résolution deux fois plus fine que celle du MNT. La plus petite distance entre ce tableau d'élévations et les élévations du MNT dans la projection du rectangle au sol donne l'estimation de l'élévation de la roue quand elle est placée sur le MNT.

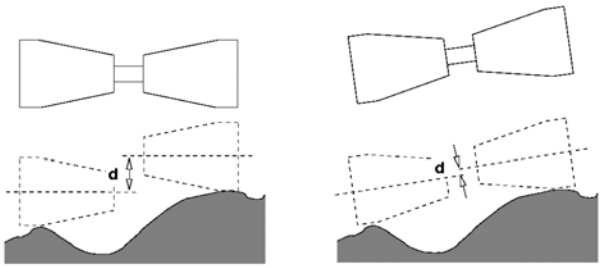


FIG. 3.11 – Principe de la fonction de placement d'essieu : `placeAxle`

Une fois les différents essieux positionnés correctement sur le MNT, l'état du robot est déterminé (attitude et configuration).
Pour le robot Lama, le positionnement absolu et relatif de ses trois essieux permet bien de connaître entièrement la configuration et la situation du robot, et donc d'associer à cette position une grandeur de *difficulté* réaliste.
La structure de Dala est différente. Ses deux essieux sont fixes dans le même plan. Ainsi en pratique nous effectuons le placement des deux essieux avec une faible tolérance d'écart entre les deux angles de roulis obtenus. Si cette différence est acceptable, le roulis attribué au robot est pris comme la moyenne des deux. Dans le cas contraire la configuration est considérée comme *non admissible*. Une fois les deux essieux placés, l'angle de tangage se détermine alors aisément (fig. 3.14). Comme pour Lama le positionnement des deux essieux permet donc de déterminer l'attitude du robot, mais il est nécessaire pour un *ATRV* de tester également les possibles *collisions de la caisse avec le sol*. C'est pourquoi cette fonctionnalité a été ajoutée à la fonction de placement du robot. Son fonctionnement s'inspire de celui de la fonction `placeWheel`. On effectue tout d'abord une projection de la caisse sur le sol (on en connaît l'orientation) pour déterminer la zone du MNT à explorer. On étudie ensuite le plus petit écart d'élévation $\delta z = (z_{caisse} - z_{MNT})$ entre le profil du bas de caisse discrétisé et les cellules du MNT. Si $\delta z \leq 0$ pour au moins une cellule, il y a collision, la configuration est donc considérée *non admissible*. Pour plus de prudence, on peut imposer un seuil $\delta_{min} > 0$. Une configuration donnée pourra alors être considérée comme admissible uniquement si $\delta z > \delta_{min}$.

Cette fonction de placement fournit ainsi :

- pour le robot Lama : une prédiction des trois paramètres de configuration $(\alpha_1, \alpha_2, \beta_3)$ (cf. figure 3.2) et des deux paramètres d'attitude (ϕ, ψ) (soit respectivement les angles de roulis et de tangage),
- pour Dala : ces deux mêmes angles d'attitude (ϕ, ψ) et une estimation du plus petit écart δz entre le bas de caisse et le sol, soit la garde au sol $gas = min(\delta z)$.

La figure 3.12 montre une comparaison entre une prévision de placement du robot Lama sur le MNT et le placement effectivement observé sur ce terrain.

(a) Prévision sur le MNT (b) Observation de la réalité terrain

FIG. 3.12 – Illustration du *posé* de robot sur un terrain accidenté : comparaison entre prédiction sur le MNT et observation.

Evaluation des Arcs Le *danger* d'une configuration est défini à partir des contraintes géométriques imposées au robot. En effet, la valeur de chacun des angles γ décrivant la configuration et l'attitude du robot doit demeurer dans un intervalle $[-\gamma_{max}; \gamma_{max}]$ admissible. Toute valeur en dehors de cet intervalle amènera à considérer cette configuration comme *non admissible*. Après normalisation ceci revient à définir un ensemble de contraintes $\{|c_i| \in [0; 1]\}$. A ces contraintes d'angles (plus la contrainte $\delta z > \delta_{min}$ pour Dala) est ajoutée une contrainte *de l'inconnu*. En effet, il peut y avoir des cellules voire des zones entières du MNT dans lesquelles aucun point 3D n'a été perçu (ceci se produit par exemple *derrière* les obstacles). D'autre part, en navigation d'extérieur, dans un souci de conservation, il est usuel (et raisonnable) de considérer que l'inconnu peut constituer un obstacle [Kelly et al., 1997]. Ainsi, un pourcentage d'inconnu nommé $pInc$, correspondant à la proportion de cellules du MNT situées sous le robot ne comportant pas d'information, est calculé lors de l'application de la fonction `placeRobot`. La contrainte qui y est associée est naturellement : $pInc \in [0; pInc_{max}]$, également normalisée[4]. Le *danger* associé à une position (x, y, θ) est alors défini comme le maximum des $(|c_i|, pInc/pInc_{max})$. Ces *dangers* sont intégrés le long de l'arc suivant les *noeuds* qui le jalonnent en prenant en compte l'abscisse curviligne et les changements de configurations successifs (un changement brutal est en effet coûteux et dangereux).

Les arcs avec leurs noeuds de configurations (positions sur le MNT) constituent un arbre à explorer. L'arc optimal est déterminé en utilisant un algorithme $A^\star$ avec comme heuristique la

[4]Nous prenons en pratique $pInc_{max} = 0.5$

distance de Dubbins qui minore la distance entre les noeuds courants et le but.

Boucle P3D et génération des consignes de vitesse Le choix d'un arc définit ainsi une trajectoire locale à suivre. Si l'on impose des vitesses maximales pour le robot on peut alors générer des consignes de vitesse (v, ω) que le robot doit suivre pour réaliser cette trajectoire. C'est le rôle du module `p3d` que de générer ces consignes de vitesse périodiquement, à destination du module de locomotion (`rflex` sur Dala, `lloco` sur Lama). L'évaluation des arcs et la sélection de celui considéré comme optimal, ainsi que la génération des consignes de vitesse constituent une tâche cyclique réalisée dans le module `p3d`. Cette tâche est exécutée chaque fois que le robot est supposé avoir parcouru la distance curviligne le séparant du prochain noeud sur l'arc utile (nous avons vu que les noeuds de configuration sur les arcs sont régulièrement espacés le long de ces derniers), ceci indépendamment de l'acquisition ou non de nouvelles données fusionnées dans le MNT. La figure 3.7 illustre l'enchaînement des tâches cycliques réalisées par le présent mode *RoughNav*.

Variance sur les angles du placement prédit Comme nous l'avons vu, chaque cellule du MNT contient l'altitude moyenne z_m des points qui ont été fusionnés dans cette cellule au cours des différentes perceptions (et qui est en pratique l'élévation exploitée pour réaliser le placement du robot), ainsi que l'*écart-type* sur cette élévation : σ_z. Nous pouvons donc associer aux prédictions d'attitude du robot (i.e. angles de roulis ψ et tangage ϕ, voire altitude z_{robot} du centre du robot) un *écart-type* ou une *variance*. Nous négligerons toutefois l'incertitude liée directement à la discrétisation du profil robot et du terrain.
Evaluer ces variances et écarts-types nous permet de prendre en compte une part de l'incertitude sur le placement du robot, dû en partie à l'incertitude sur le MNT construit. Ces données seront utilisés en pratique pour effectuer des comparaisons de ces données incertaines avec d'autres estimations de ces angles d'attitude (voir chapitre 5). Ils pourraient également être exploités lors de l'évaluation des coûts sur les arcs par `p3d` mais cela ne nous semble pas nécessaire à une bonne planification et ajouterait une charge de calcul trop importante.

Considérons le placement d'un essieu et notons z_r et z_l les altitudes prévues des points de contact respectivement des roues droite et gauche (figure 3.13). Tout d'abord, afin de prendre en compte l'influence des écarts-types de toute la zone du MNT se trouvant sous une roue (car c'est bien toute cette zone qui conditionne la position du point de contact d'une roue avec le sol et l'élévation qui y est finalement attribuée), nous considérons l'écart-type de l'élévation au point de contact d'une roue, respectivement σ_r écart-type de z_r pour la roue droite et σ_l écart-type de z_l pour la roue gauche, comme la moyenne des écarts-type des cellules de la zone du MNT sous la roue.

L'angle de roulis pour le placement d'un essieu est :

$$\psi = \arctan\left(\frac{z_l - z_r}{d}\right)$$

où d est la distance euclidienne entre les projections au sol des deux points de contact roue-sol, de coordonnées (x_r, y_r) et (x_l, y_l), soit : $d = \sqrt{(x_l - x_r)^2 + (y_l - y_r)^2}$.
Dans un cas général, la matrice de covariance du vecteur $\boldsymbol{\psi} = f(\boldsymbol{z})$ est $\Psi = FZF^T$, où F est la matrice jacobienne de la fonction f et Z la matrice de covariance de $\boldsymbol{z}$.
Ici,

$$F = \frac{\partial \psi}{\partial \boldsymbol{z}} = [F_r \, F_l]$$

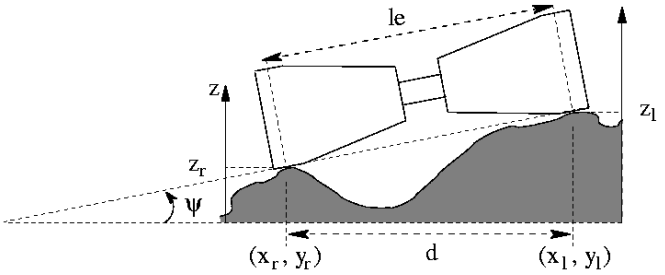


FIG. 3.13 – Vue de face d'un essieu après placement, montrant les élévations z_r et z_l de chaque côté. La longueur *le* est la longueur entre les points de contact. Les points de coordonnées (x_r, y_r) et (x_l, y_l) dans la grille sont les projections sur le sol des points de contact respectivement de la roue droite et de la roue gauche avec le sol. d est la distance euclidienne entre ces deux points.

avec :

$$F_r = \frac{\partial \psi}{\partial z_r^T} \quad \text{et} \quad F_l = \frac{\partial \psi}{\partial z_l^T}$$

Dans le MNT, les élévations z_r et z_l associées aux deux points distants (x_r, y_r) et (x_l, y_l) de la grille discrétisée peuvent être considérées comme indépendantes du moment que les cellules auxquelles elles appartiennent sont bien distinctes (ce qui est toujours le cas vues les dimensions du robot par rapport à l'échelle du MNT). Ainsi, la matrice de covariance de $\boldsymbol{z} = [z_r\, z_l]^T$ peut s'écrire :

$$Z = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_l^2 \end{bmatrix}$$

La matrice de covariance de ψ est ainsi égale à :

$$\Psi = F_r \sigma_r^2 F_r^T + F_l \sigma_l^2 F_l^T$$

Cette expression ne faisant intervenir ici que des scalaires :

$$\sigma_\psi{}^2 = F_r^2 \sigma_r^2 + F_l^2 \sigma_l^2$$

Or, dans le cas présent,

$$F_l = -F_r = \frac{\partial}{\partial z_l}\left(\arctan\left(\frac{z_l - z_r}{d}\right)\right) = \frac{d}{d^2 + (z_l - z_r)^2}$$

d'où :

$$\sigma_\psi{}^2 = (\sigma_r^2 + \sigma_l^2)\frac{d^2}{(d^2 + (z_l - z_r)^2)^2}$$

Nous l'avons vu, la fonction `placeWheel`, appelée à plusieurs reprise par `placeAxle` pour déterminer quel angle de roulis va être attribué à l'essieu placé, exploite des discrétisations du profil de la roue. Afin de gagner en temps de calcul, des profils discrétisés sont préparés lors de l'initialisation, avec différentes orientations de la roue, en prenant un pas δ. Ceci revient à une discrétisation de l'angle de roulis ψ. Il faut donc ajouter la variance due à cette discrétisation. On montre que l'écart-type correspondant vaut :

$$\sigma_d = \frac{1}{\delta}\int_{-\delta/2}^{\delta/2} x^2 dx = \frac{\delta}{2\sqrt{3}}$$

La variance totale sur ψ devient alors :

$$\sigma_{\psi_d}{}^2 = \sigma_\psi{}^2 + \sigma_d{}^2$$

Pour ne pas alourdir les notations, nous noterons par la suite $\sigma_{\psi_i}{}^2$ cette variance totale pour l'essieu i, faisant fi de la spécification de l'étape de discrétisation.

Un angle de roulis prédit ψ est obtenu pour chaque essieu du robot. Les positions relatives en (x, y) où les placements des différents essieux du robot sont effectués sont bien entendu dépendantes de la structure du robot, mais les différents angles de roulis qui résultent du placement de chacun de ces essieux sont indépendants (car ils sont sur des zones distinctes du MNT). Par contre, si la différence entre deux de ces angles trouvés est trop importante pour être supportée par la structure mécanique réelle du robot, la position sera considérée comme non praticable, et donc rejetée. La variance du roulis prédit peut être aisément obtenue à partir de celles des essieux placés. Ainsi, pour un robot tel que Lama l'angle de roulis du robot est simplement celui de l'essieu central. Pour un robot à structure rigide tel que Dala, le roulis pourra être considéré comme la moyenne des roulis prédits obtenus lors du placement des deux essieux avant et arrière, si toutefois la configuration prédite est considérée comme acceptable. Ainsi, en notant ψ_{Av} l'angle de roulis obtenu lors du placement de l'essieu avant et ψ_{Ar} celui pour l'essieu arrière, ψ_{Av} et ψ_{Ar} étant indépendants, la variance de l'angle de roulis attribué au robot est : $\sigma_\psi{}^2 = (\sigma^2_{\psi_{Ar}} + \sigma^2_{\psi_{Av}})/4$.

La variance sur l'angle de tangage ϕ peut se calculer d'une manière similaire si l'on considère que l'écart-type de l'élévation du centre de la roue gauche (resp. la roue droite) est égal à σ_l (respectivement σ_r). L'élévation du centre de l'essieu étant $z_{essieu} = (z_r + z_l)/2$ (avec $essieu = Av$ ou Ar) et z_r et z_l pouvant être considérés comme indépendants, la variance sur z_{essieu} s'exprime par :

$$\sigma_{z_{essieu}}{}^2 = (\sigma_r^2 + \sigma_l^2)/4 \tag{3.1}$$

Considérons le cas d'un robot ayant une structure semblable à celle de Dala. Le placement repose sur deux essieux, l'avant (Av) et l'arrière (Ar). Le placement de chacun de ces deux essieux nous donne z_{Av} et z_{Ar}, et donc aussi leurs variances respectives σ_{Av} et σ_{Ar}, selon l'équation 3.1. L'angle de tangage du robot s'exprime, d'après la figure 3.14 :

$$\phi = \arcsin\left(\frac{z_{Ar} - z_{Av}}{L}\right)$$

où L est la distance inter-essieux. On en déduit donc la variance de ϕ :

$$\sigma_\phi{}^2 = (\sigma_{Av}^2 + \sigma_{Ar}^2)\frac{1}{L^2 - (z_{Ar} - z_{Av})^2}$$

3.2.3 Suivi de chemins

Ce mode de navigation, mis au point lors de travaux précédents au laboratoire [Aviña, 2005], est destiné aux environnements *semi-structurés*. Il exploite (et nécessite) la présence de voies de type *chemin* dans l'environnement, repérées grâce à une caméra couleur. En effet ces voies sont conçues pour faciliter le déplacement d'un véhicule à roues, il est donc intéressant de profiter de ces aménagements existants dans le cadre d'une navigation de robot terrestre en terrain a priori inconnu. L'extraction du chemin est réalisée grâce à une classification basée sur des attributs de couleur et de texture. Une technique hybride de segmentation couleur permet d'extraire des régions dans l'image pour lesquelles sont calculés des attributs de couleur, de texture et de

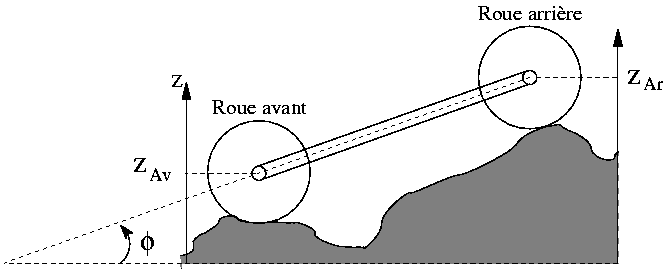


FIG. 3.14 – Angle de tangage ϕ de Dala prédit par le placement : il est déduit de la différence entre les deux élévations z_{Av} et z_{Ar}.

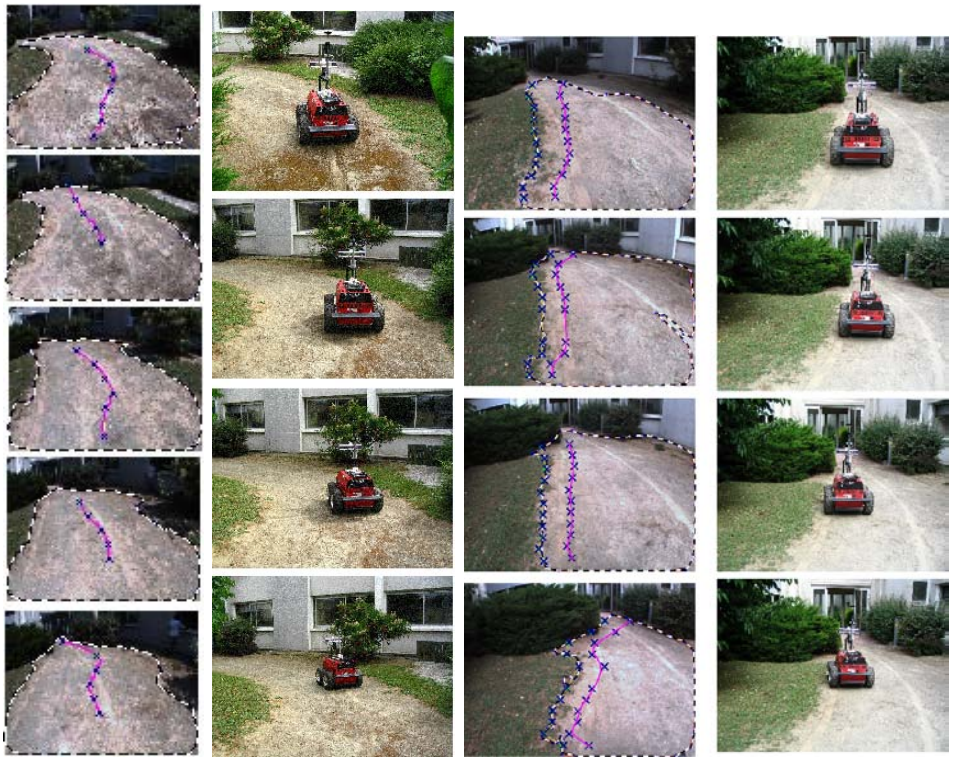
contexte. Une méthode de classification probabiliste reposant sur un apprentissage supervisé permet d'étiqueter les zones suivant des types particuliers (tels que *Arbre, Ciel, Herbe, Chemin*). Un module de navigation visuelle exploite alors cette description de la scène perçue par la caméra couleur pour réaliser une primitive de déplacement élémentaire telle que le *suivi de chemin* ou le *suivi de bordure*. Pour cela, les contours échantillonnés et filtrés des régions *Chemin* sont modélisés par des *splines* cubiques, et des points formant la trajectoire à suivre sont estimés, liés ensuite par des courbes de Bézier (figure 3.15). Les éléments de cette trajectoire permettent alors de générer des commandes de mouvement pour le robot sous la forme de consignes (v, ω) (voir [Mateus et al., 2005] pour plus de détails). Il convient de noter que cette fonctionnalité de navigation utilise une *hypothèse de sol plat*, restreignant ainsi encore davantage son contexte d'application.

3.2.4 Résumé des caractéristiques des modes

Le tableau suivant résume les principales caractéristiques des modes de navigation présentés. Précisons quelques définitions de termes que nous y employons :

- Nous définissons le *contexte valide* d'un mode comme le contexte (type d'environnement) qu'il est *nécessaire* d'avoir pour que ce mode fonctionne correctement (exemple : le terrain *doit* être plat pour pouvoir utiliser le mode *FlatNav*).
- Le *contexte dédié* est le contexte pour lequel le mode a été pensé et conçu. Il s'agit donc du contexte dans lequel ce mode est *supposé se comporter le mieux*, selon son concepteur. Il s'agit généralement d'une éventuelle restriction du *contexte valide*.

Nous rappelons également dans ce tableau les robots sur lesquels chacun de ces modes est disponible.



(b) Suivi de chemin (c) Suivi de bordure

FIG. 3.15 – Modalité de suivi visuel de chemin ou de bordure avec le robot Dala (d'après [Aviña, 2005])

Mode	**FlatNav**	**RoughNav**	**Suivi chemin**
Robots	Dala	Dala & Lama	Dala
Contexte valide	Terrain plat	Tous	Présence de chemins plats
Contexte dédié	id.	Terrain accidenté en environnement naturel	Environnement semi-structuré (avec présence de *chemins* globalement *plats*)
Avantages principaux	*Vitesses rapides* possibles	Exploitation de données *denses* sur le terrain - Méthode générique (exploitable sur presque tout type de terrain d'extérieur)	Exploitation de structurations présentes dans l'environnement pour faciliter la circulation de véhicules
Inconvénients principaux	Limitation de contexte importante - Détection des obstacles seulement dans un plan	*Vitesses lentes* nécessaires (temps de calcul)	Limitation de contexte importante - Vitesses lentes (temps de calcul)

3.3 Les modes de locomotion

Nous proposons dans cette partie quelques exemples de modes de locomotion que nous exploitons. Le rôle d'un mode de *locomotion* est de faire en sorte que le robot réalise effectivement le mouvement demandé par un mode de navigation, par exemple assurer de suivre les consignes (v, ω) envoyées par le mode de navigation actif, en effectuant les actions adaptées sur les capacités de déplacement du robot, à savoir les mobilités internes actives (moteurs de roues et éléments de suspension active si le robot en dispose).

3.3.1 Roulement pur

Dans notre cas, il s'agit de réaliser une commande de rovers de type "char" par un couple de consignes en vitesse : (v, ω). Cette commande peut être éventuellement rendue plus proche de la réalité du terrain grâce à une méthode telle que le TALC, proposée initialement dans [Peynot, 2002] puis améliorée dans [Peynot et al., 2003], que nous présentons ici brièvement.

Commande des "skid-steering" rovers

Les robots Dala et Lama sont des robots de type *skid-steering*, autrement dit de type "char" : ils ne disposent pas de roues directionnelles. De façon naturelle, un tel châssis se commande en vitesse. Prenons l'exemple de Lama [Peynot, 2002]. L'indépendance des roues permet, au niveau d'un essieu isolé, d'obtenir très simplement un mouvement de rotation grâce à l'application de vitesses *différentes* sur les roues droite et gauche (V_r à droite et V_l à gauche). Ainsi, un essieu exécute à chaque instant une trajectoire *circulaire* autour d'un point fixe : le centre instantané de rotation. La modification dynamique des vitesses permet de changer la position de ce point fixe et d'exécuter tous types de trajectoires compatibles avec la structure du châssis.

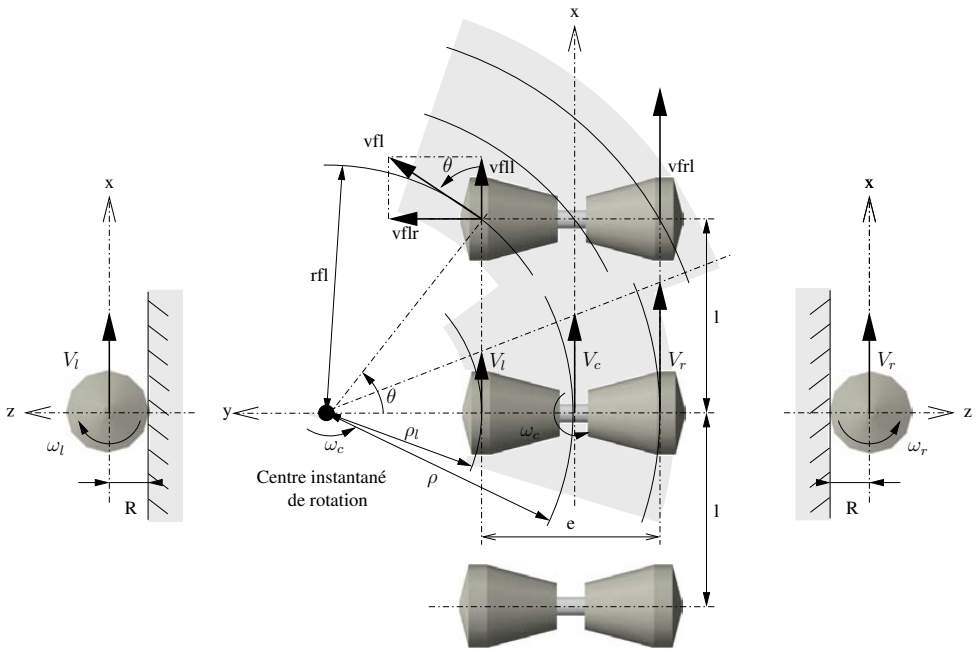


FIG. 3.16 – Commande des rovers de type *skid-steering*, d'après [Mallet, 2001].

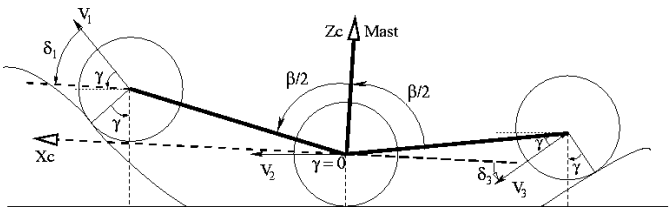
La figure 3.16 illustre ce principe pour une structure de type Marsokhod [Mallet, 2001] : deux vitesses ω_r et ω_l sont appliquées aux roues gauche et droite de l'essieu central, provoquant la rotation du châssis à la vitesse angulaire $\omega_c = \rho V_c$, où ρ est le rayon instantané de giration. L'hypothèse de roulement sans glissement permet d'exprimer la vitesse du centre du robot en fonction des vitesses des roues centrales, de leur rayon R et de l'entraxe e (distance entre les points de contact des deux roues) :

$$V_c = R(\omega_r + \omega_l)/2 \quad (3.2)$$
$$\omega_c = R(\omega_r - \omega_l)/e \quad (3.3)$$

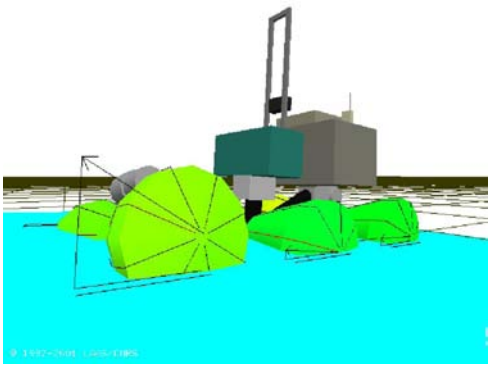
Le différentiel de vitesses sur les roues centrales gauche et droite provoquant une rotation de cet essieu central, il est alors nécessaire d'appliquer sur les essieux avant et arrière des vitesses compatibles avec ce mouvement pour autoriser la rotation globale du châssis. En pratique, la commande classique d'un robot de ce type applique *la même vitesse de rotation sur toutes les roues d'un même côté* : $\omega_r = V_r/R$ pour toutes les roues droites et $\omega_l = V_l/R$ pour toutes les roues gauches (en supposant un roulement sans glissement, et en notant R le rayon commun des roues). Ce raisonnement, appliqué ici à une structure multi-essieux de type Marsokhod (Lama), peut s'appliquer de manière similaire à Dala.

Commande adaptée au terrain

La méthode que nous venons de voir ne prend pas en compte la forme du terrain. En réalité, l'application directe de ω_r et ω_l provoque le glissement de certaines roues sur les terrains accidentés. Nous avons proposé une méthode de commande permettant d'éviter ce problème en prenant en compte explicitement la forme du terrain sur lequel le robot évolue [Peynot et al., 2003]. La démonstration sera effectuée à partir de la structure des robots de type *Marsokhod* (Lama), ayant un châssis articulé intéressant pour ce genre de problèmes. En effet, ses trois essieux articulés avec ses six roues indépendantes permettent d'avoir plus d'informations (et de la redondance). Cependant, une méthode semblable peut également être mise en oeuvre avec d'autres types de châssis.



(a) Vitesses linéaires des trois roues du côté gauche de Lama



(b) Représentation 3D de Lama avec les vitesses linéaires de chaque roue et leur projection sur le plan $(X_c Y_c)$. La roue avant-gauche est en train de franchir un rocher.

FIG. 3.17 – Illustration des vitesses linéaires du robot Lama sur terrain accidenté

Ce phénomène est illustré en 2D et en 3D sur la figure 3.17 : les normes des trois vecteurs vitesse linéaire sont les mêmes mais leurs projections dans le plan de l'essieu central (X_cY_c), considéré localement comme le plan de déplacement du robot, sont bien différentes. Comme les trois roues font partie du même solide (le robot), certaines roues sont amenées à glisser. Pour chaque roue, *l'angle entre sa vitesse linéaire et le sol* doit être pris en compte pour éviter les glissements. En conséquence, l'objet de notre commande "adaptée terrain" (TALC : *Terrain Adaptive Locomotion Control*) est de faire en sorte que les vitesses linéaires *réelles* de toutes les roues du robot respectent les relations cinématiques qui les lient, du fait de leur appartenance mécanique au même solide. Si cette condition est respectée à chaque instant, la TALC ne devrait pas *provoquer* de situation de glissement. Bien entendu, des glissements se produiront encore, mais la TALC permet de les réduire. Ainsi, par exemple, si une roue avant monte sur une pierre alors que les autres restent sur du terrain plat (cf. figure 3.17), une vitesse plus importante est appliquée à la roue en question pour compenser l'influence du nouvel angle entre sa vitesse linéaire et le plan de déplacement.

Réalisation Grâce aux capteurs proprioceptifs, la configuration et l'attitude du robot sont connues à chaque instant. Connaissant ces données, leurs dérivées et les vitesses de référence, nous estimons la direction et la norme de la vitesse linéaire de chaque roue. Ces normes nous donnent alors la vitesse de référence à appliquer à une roue donnée : $\omega_{wheel} = V_{wheel}/R$.

Repères Il nous faut calculer les vitesses linéaires de chaque roue par rapport au sol (repère $\Re_G$). Un autre repère utilisé est le repère dit *de l'essieu central* : $\Re_c$ (voir figure 3.18). Son centre C est le centre de l'essieu central du robot, l'axe Y est l'essieu orienté du côté droit vers le gauche et l'axe Z est le long du mât du robot, orienté vers le haut. Le robot avance toujours selon l'axe X de $\Re_c$ (X_c) et sa rotation $\omega = \omega_{Gyro}$ s'effectue et est mesurée dans le plan (X_cY_c).

Deux autres repères sont introduits : $\Re_l$ et $\Re_r$. $\Re_l$ est le résultat d'une translation de $\Re_c$ le long de son axe Y telle que le centre C_l de $\Re_l$ est le centre de la roue gauche de l'essieu central. Symétriquement, $\Re_r$ est lié à la roue droite de l'essieu central.

Comme nous l'avons dit, les mouvements du robot sont définis par deux vitesses de références : V_r pour le côté droit et V_l pour le gauche. Ainsi, le but du TALC est que la composante en X_c de la vitesse de $\Re_l$ (respectivement $\Re_r$) soit égale à V_l (resp. V_r).

Calcul des consignes de vitesse corrigées Considérons un centre de roue nommé B, sur le côté s (r si droite, l si gauche). Premièrement, $\vec{V}_{B/\Re_c}$, la vitesse de B par rapport au repère $\Re_c$, n'est due qu'aux variations de configuration du robot, connues grâce aux capteurs mesurant les angles de configuration et d'attitude (α_1, α_2, β_3, ϕ et ψ).
Deuxièment, la vitesse de B par rapport au repère sol $\Re_G$ peut être exprimée de la manière suivante :

$$\vec{V}_{B/\Re_G} = \vec{V}_{B/\Re_s} + \vec{V}_{C_s/\Re_G} + \overrightarrow{BC_s} \wedge \vec{\Omega}_{\Re_s/\Re_G}$$

Du fait du lien entre $\Re_s$ et $\Re_c$, $\vec{\Omega}_{\Re_s/\Re_G} = \vec{\Omega}_{\Re_c/\Re_G}$ et $\vec{V}_{B/\Re_s} = \vec{V}_{B/\Re_c}$. En conséquence :

$$\vec{V}_{B/\Re_G} = \vec{V}_{B/\Re_c} + \vec{V}_{C_s/\Re_G} + \overrightarrow{BC_s} \wedge \vec{\Omega}_{\Re_c/\Re_G}$$

Le TALC impose que $|\vec{V}_{C_s/\Re_G}|_{X_c}$ soit égal à V_s.
$|\vec{V}_{C_s/\Re_G}|_{Y_c} = 0$, vu que Y_c contient le centre instantané de rotation.
Enfin, $|\vec{V}_{C_s/\Re_G}|_{Z_c} = \dot{z}_{C_s}$ peut être calculé en utilisant la relation entre C_s et C :

$$\vec{V}_{C_S/\Re_G} = \vec{V}_{C/\Re_G} + \overrightarrow{C_sC} \wedge \vec{\Omega}_{Rc/\Re_G}$$

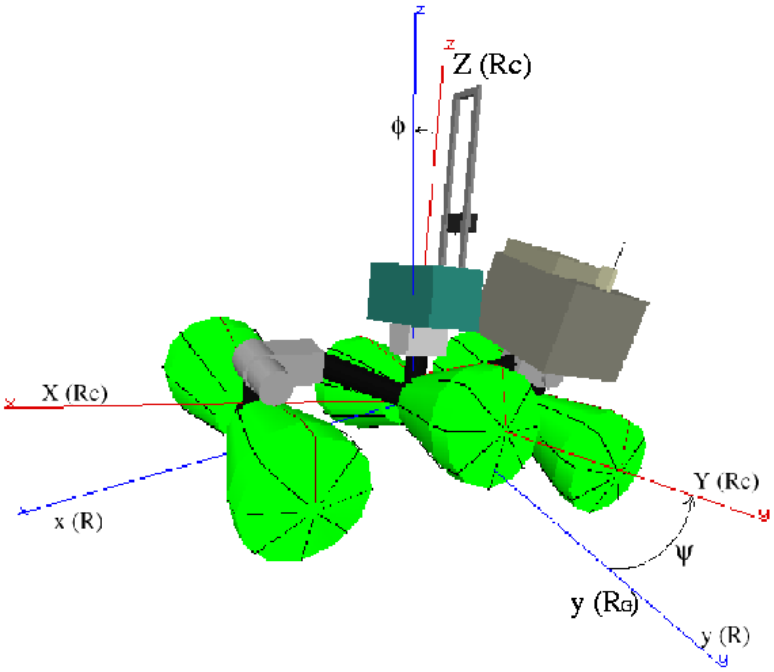


FIG. 3.18 – Le repère $\Re_c$ par rapport au sol ($\Re_G$).

et l'estimation de $|\vec{V}_{C/\Re_G}|_{Z_c}$ réalisée grâce aux capteurs proprioceptifs.
Ainsi, en considérant l'expression de $\vec{\Omega}_{\Re_c/\Re_G}$, la vitesse de B peut finalement être exprimée dans $\Re_c$ comme :

$$\vec{V}_{B/\Re_G} = \vec{V}_{B/\Re_c} + \begin{pmatrix} V_s \\ 0 \\ \dot{z}_{C_s} \end{pmatrix} + \overrightarrow{BC_s} \wedge \begin{pmatrix} \dot{\psi} \\ \dot{\phi} \\ \omega \end{pmatrix}$$

Si les consignes globales de vitesse du robot sont respectées, la vitesse de B devrait donc être : $\vec{V}_B = (\dot{x}\,\dot{y}\,\dot{z})^T$, calculé ci-dessus. Comme $\dot{y}$ est négligeable vis à vis de $\dot{x}$ et $\dot{z}$, particulèrement pour des robots lents comme Lama, la norme $V_B = \sqrt{\dot{x}^2 + \dot{z}^2}$ donne la vitesse qui doit être appliquée à la roue : $\omega_{wheel} = V_B/R$ (roulement sans glissement).

Finalement, toutes les consignes de vitesse "adaptées" sont calculées pour que la projection de la vitesse de l'essieu central sur (X_cY_c) soit V_l sur le côté gauche et V_r sur le côté droit et que les relations cinématiques entre les roues soient respectées.

Estimation des angles de contact Si δ_B est l'angle de $\vec{V}_B$ par rapport au plan (X_cY_c) (en direction du déplacement), $\cos(\delta_B) = \dot{x}/V_B$. Comme l'orientation de $\Re_c$ par rapport au sol est connue à chaque instant (par l'intermédiaire des inclinomètres), l'angle γ_B de la vitesse par rapport à l'*horizontale* peut être estimé. Cet angle est aussi l'angle de contact de la roue avec le sol. Le TALC fournit donc une estimation de tous les angles de contact des roues avec le sol : γ_{wheel}. Cette information permet de faire des comparaisons cohérentes des différentes vitesses et sera utilisé pour le monitoring de locomotion (section 5.2). Notons que l'estimation est basée uniquement sur les mesures des angles de configuration et d'attitude et les vitesses désirées ω_{ref} et V_{ref}, contrairement à la méthode proposée dans [Iagnemma et al., 2000a, Iagnemma et al., 2000b], qui

utilise toutes les mesures de vitesses de roues. Ainsi, tant que le déplacement global du robot respecte les références, les estimations d'angles de contact restent correctes pour *toutes* les roues, même si *certaines* dérapent.

Résultats La figure 3.19 illustre les corrections de vitesse générées par le TALC lors d'un test simple : le robot Lama se déplace en ligne droite à une vitesse de 5 cm/s, le terrain est globalement plat, hormis une pierre sur la trajectoire du côté gauche du robot. Chacune des roues gauches monte donc sur la pierre (puis redescend), successivement. Pour compenser l'influence de la forme du terrain, la vitesse élémentaire de chaque roue est augmentée puis rediminuée au passage sur la pierre.

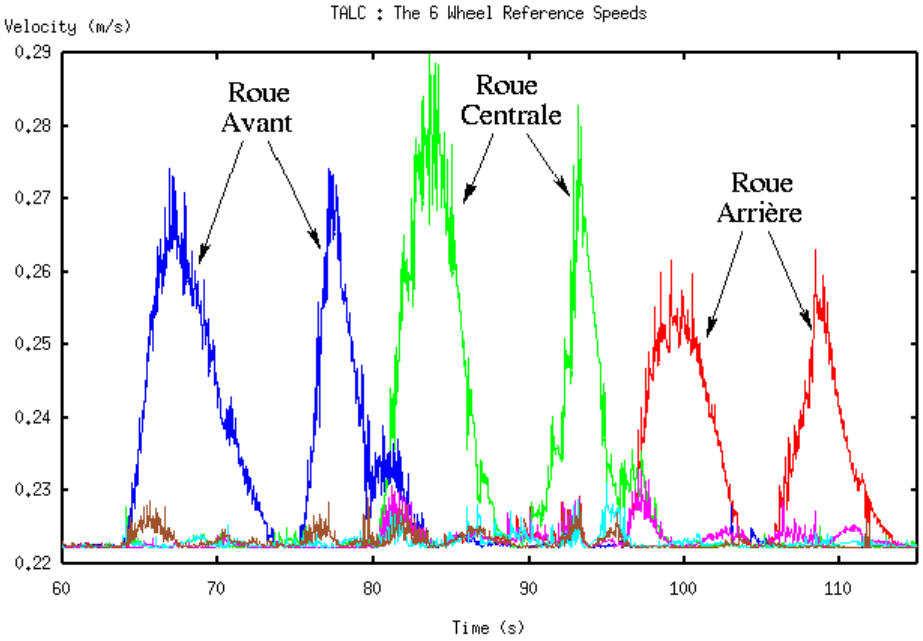


FIG. 3.19 – Les 6 vitesses de référence générées par le TALC pendant un simple test consistant à faire passer le robot sur un terrain comportant une pierre isolée sur son côté gauche.

Notons enfin qu'une commande de ce type peut également être mise en oeuvre sur un robot tel que Dala. Néanmoins, le contrôle bas niveau des moteurs des roues a été réalisé par la société qui a conçu les ATRV, *iRobot*, et le code source correspondant n'est pas accessible pour un tiers. Nous nous contentons donc en pratique de leur intégration d'origine.

3.3.2 Péristaltisme

Ce mode de locomotion, possible par exemple sur Lama (et aussi sur Hylos 2, voir annexe B), vise à faire "ramper" le robot à la manière d'une chenille, en exploitant les mobilités internes des robots qui en disposent. Pour un robot tel que Lama, il s'agit de faire avancer de manière cyclique les essieux un par un, successivement, en commençant par l'essieu avant (figure 3.20). Cela nécessite un actionnement coordonné des articulations actives et des moteurs des roues. Ce mode de déplacement permet d'obtenir une traction plus efficace dans des terrains peu cohésifs (constitués de sable ou de graviers par exemple) dans lesquels le roulement simple peut se révéler inefficace. Toutefois, il présente également d'importantes limitations : il ne génère que des

déplacements *lents* et principalement en *ligne droite*[5]. Dans la plupart des cas il ne pourra donc pas accepter des consignes sous une forme classique (v, ω) fournis par les modes de navigation habituels si $\omega \neq 0$, comme c'est le cas pour les autres modes de locomotion. Par conséquent, ce mode demande un traitement particulier de la part de la couche de navigation.

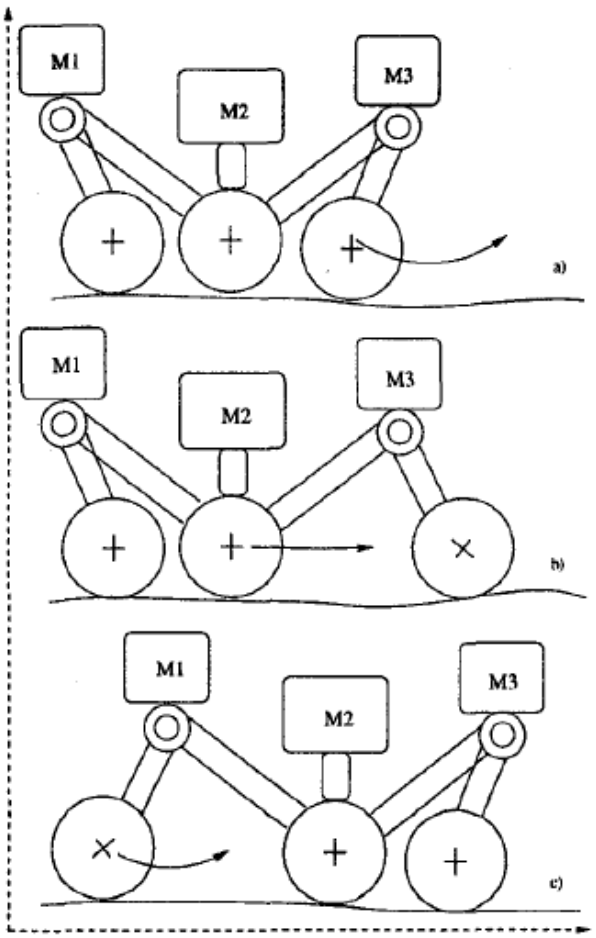


FIG. 3.20 – Illustration du mode de locomotion *péristaltique* pour un châssis de type Marsokhod comme Lama, d'après [Andrade et al., 1998].

3.3.3 Résumé des caractéristiques des modes

Les tableaux suivants résument les principales caractéristiques des modes de locomotion présentés. Nous employons les mêmes termes que pour le tableau concernant les modes de navigation (section 3.2.4).

[5] si les mobilités internes actives droite/gauche sont indépendantes (comme sur Hylos 2 par exemple, voir annexe B) des rotations peuvent être envisagées dans un mode péristaltique *dissymétrique*.

Mode	**Roulement**	**Péristaltisme**
Robots	Dala & Lama	Lama
Contexte valide	Tous	Tous
Contexte dédié	Terrain cohésif	Terrain peu cohésif
Avantages principaux	Vitesses rapides possibles - Méthode générique de mise en oeuvre simple	Constitue une alternative efficace au roulement classique sur les terrains glissants
Inconvénients principaux	Peut générer d'importants glissements	Vitesses très lentes - Trajectoires nécessairement en lignes droites

3.4 Exemples de systèmes multi-modes

Nous proposons des exemples de systèmes multi-modes exploitant les modes de navigation et de locomotion qui viennent d'être décrits dans ce chapitre, destinés aux robots Dala et Lama, et sous la forme qui a été précisée dans le chapitre 2. De part la nature de ces robots et leur équipement, nous verrons deux variantes d'un système *multi-modes de navigation* pour Dala, et une proposition de système *multi-modes de locomotion* pour les robots à mobilités internes actives tels que Lama. Un autre exemple prévu pour le robot Hylos 2 peut également être vu en annexe B.

3.4.1 Locomotion de Lama

Le système multi-modes de locomotion pour Lama est illustré par la figure 3.21. Il repose sur les modes de locomotion *Roulement* (assorti de la commande TALC) et *Péristaltisme*. Le premier est le mode classique par défaut qui est le plus couramment utilisé. Le deuxième sera préféré si le robot rencontre un terrain non cohésif sur lequel il pourra avoir des difficultés de locomotion par roulement classique.

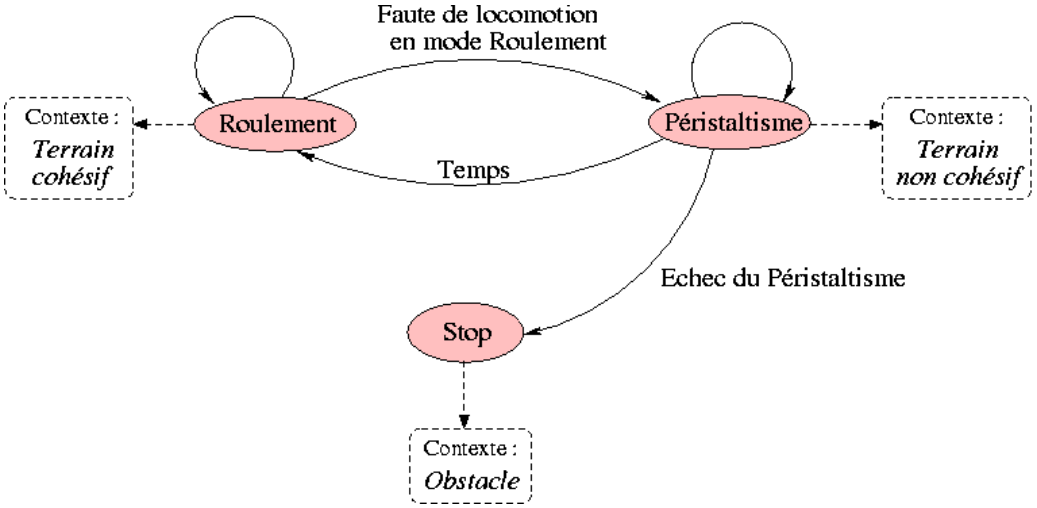


FIG. 3.21 – Système multi-modes de locoomotion pour Lama, exploitant les modes *Roulement* (plutôt dédié aux terrains cohésifs) et *Péristaltisme* (dédié aux terrains peu cohésifs).

3.4.2 Navigation de Dala

Le système multi-modes de navigation pour Dala est illustré par la figure 3.22. Il repose pour sa part sur les deux modes de navigations *FlatNav* et *RoughNav*. Comme nous l'avons vu, les contextes dédiés de ces modes sont respectivement les terrains plats et accidentés. Le mode *FlatNav* est à favoriser sur terrain plat car le robot peut se déplacer plus vite. Sans information de comportement fournies par les moniteurs, le mode à appliquer est donc uniquement fonction de la perception de ce contexte.
Cependant, nous commençons à voir sur la figure les causes qui favoriseront les transitions pour inciter au changement de mode. En effet, avec les moniteurs, si le mode initial est *FlatNav*, tout événement laissant à penser que le terrain n'est pas facile et plat aura tendance à augmenter la probabilité que le mode à appliquer soit plutôt *RoughNav*, plus "générique" et "prudent". Concernant les transitions issues de *RoughNav*, ce sont les événements révélant un échec de ce mode qui amèneront dans le mode *Stop* (correspondant au contexte "obstacle" c'est à dire plus généralement terrain non franchissable par le robot), si le contexte n'est pas favorable à une reprise du mode *FlatNav*.

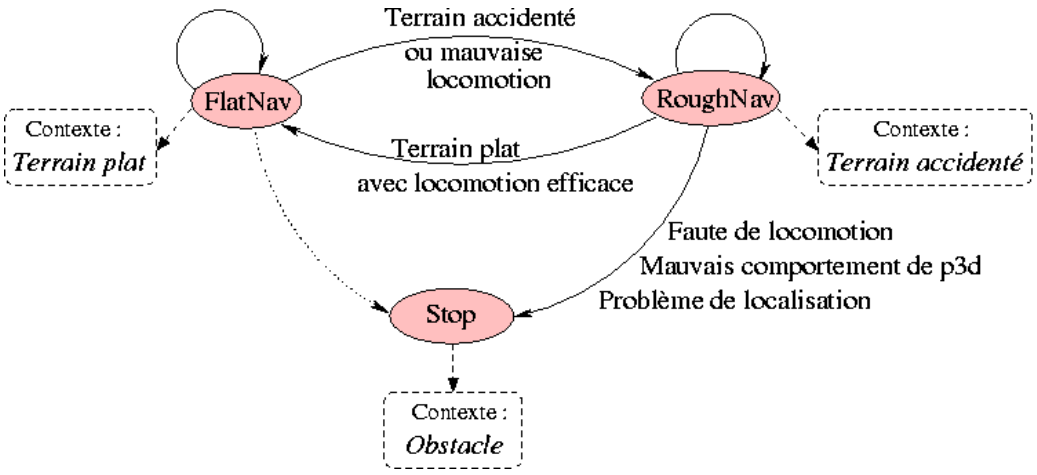


FIG. 3.22 – Système multi-modes de navigation pour Dala, exploitant les modes dits *FlatNav* (valide seulement sur terrain plat) et *RoughNav* (dédié aux terrains accidentés).

Une variante du précédent système exploitant le mode *Suivi de chemin* est proposée en figure 3.23. Le contexte d'application de ce mode est un environnement semi-structuré avec présence de chemin (plat) en direction du but à atteindre.
Vu que la procédure de classification permettant d'extraire un chemin dans la scène est probabiliste, on peut avoir une notion de niveau de "discrimination" de ce chemin. Si ce niveau n'est pas suffisant, il conviendra de revenir à un mode plus "classique" tel que *FlatNav*.

3.4.3 Locomotion d'Hylos 2

Un autre type de plate-forme et de modes, pour lesquels une étude préliminaire a été réalisée en extension de ces travaux, est présenté dans l'annexe B. Cet autre exemple est intéressant car la structure du robot Hylos 2, munie de multiples mobilités internes actives, peut permettre la mise au point et l'exploitation d'une variété plus importante de modes de locomotion en comparaison avec une plate-forme telle que Lama.

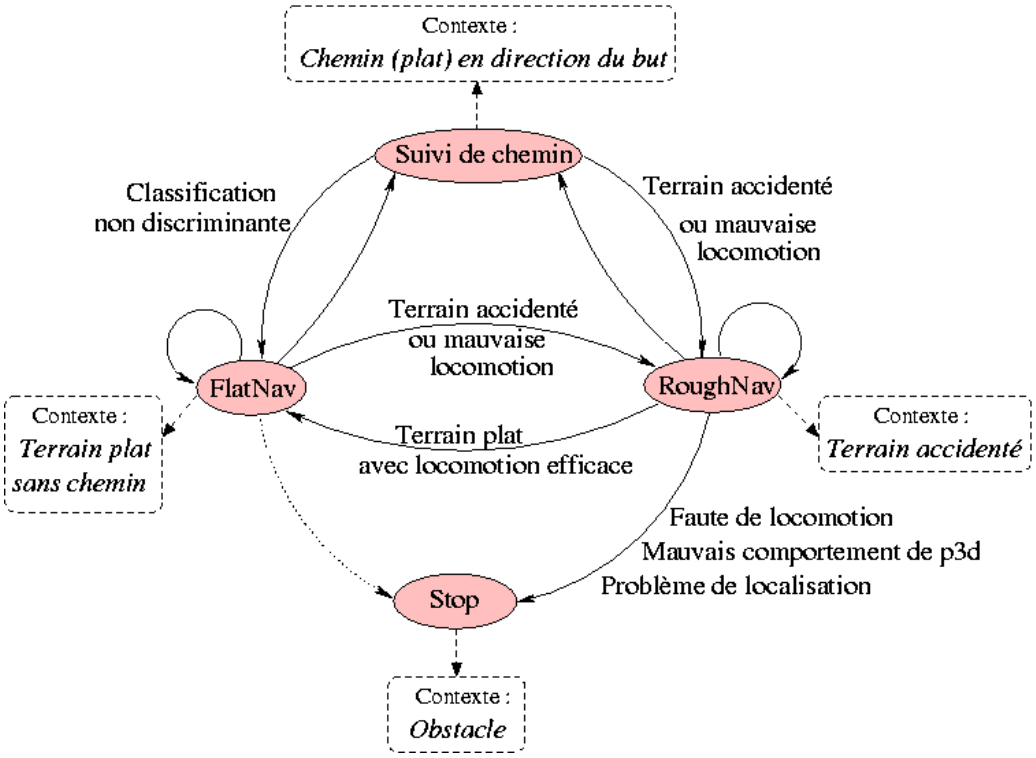


FIG. 3.23 – Système multi-modes de navigation pour Dala, exploitant les modes dits *FlatNav*, *RoughNav* et *Suivi de chemin*.

3.5 Conclusion

Nous avons présenté dans ce chapitre nos plates-formes d'expérimentation puis les différents modes de déplacement que nous exploitons dans ces travaux. Leur présentation a été séparée suivant leurs deux composantes principales : *navigation* et *locomotion*, du fait de leurs différences de fonctionnalité et de finalité. Nous avons pu voir enfin comment ces modes peuvent s'intégrer dans le formalisme qui a été vu dans le chapitre précédent, en proposant des systèmes multi-modes de locomotion et de navigation pour nos deux robots.

Nos contributions dans ce chapitre concernant les modes de déplacement sont :

- l'adaptation du mode *RoughNav* au robot Dala (à partir d'une version existant déjà pour le robot Lama), avec en particulier l'ajout d'un test de garde au sol dans la fonction de placement et dans celle de coût,
- une évaluation de l'incertitude liée aux résultats du placement du robot, à savoir les angles d'attitude ϕ et ψ ainsi que la garde au sol gas, sous la forme d'écarts-types σ_ϕ, σ_ψ et σ_{gas},
- une commande en vitesse pour un robot de type *skid-steering* pour une locomotion qui s'adapte au relief du terrain et permet également une estimation des angles de contact des roues avec le sol.

Bien d'autres modes de déplacement ainsi que d'autres types de plates-formes pourraient être ajoutés à ceux évoqués ici et être avantageusement intégrés dans les systèmes multi-modes. Cependant, il ne s'agit pas seulement de les intégrer et les utiliser, il s'agit également d'avoir une connaissance suffisamment grande et précise de leur fonctionnement (comme c'est le cas pour les modes vus ici) pour être capable de bien définir leur *contexte* d'utilisation et de mettre au point des *moniteurs* capables d'évaluer leur comportement.

Ces deux derniers points font respectivement l'objet des deux chapitres suivants.

Chapitre 4

Le contexte : Données terrain

L'un des éléments fondamentaux du formalisme de sélection de mode est le *modèle d'observation du terrain*, qui permet de fournir les informations de *contexte* guidant l'estimation du mode de déplacement à appliquer. Ces observations reposent sur un certain type de représentation de l'environnement, que nous proposons dans ce chapitre.

Comme nous avons pu en voir un aperçu dans le chapitre 1, de nombreux types de représentation de l'environnement pour un robot mobile ont été proposés dans la littérature ces dernières années, la plupart visant principalement à identifier l'espace libre et les obstacles. Dans le cadre de la robotique en environnement naturel, une représentation plus riche est nécessaire, afin de prendre en compte le relief du terrain et étudier la possibilité pour un robot de traverser des zones de terrain accidenté. Parmi ces méthodes, celles qui nous paraissent les plus pertinentes sont celles réalisant ce que nommerons de la *perception par simulation d'action*. Il s'agit d'étudier la situation dans laquelle se retrouverait un objet (par exemple le robot) s'il était placé à un endroit donné de l'environnement, autrement dit de simuler une interaction de l'objet avec l'environnement perçu. C'est ce type de représentations que nous exploitons pour générer notre *modèle d'observation du terrain*.
Nous présentons dans la section suivante deux antécédents notables dans ce genre de méthodes, utilisées généralement pour la navigation de rovers planétaires, à savoir les méthodes du CNES et du JPL de la NASA pour le programme MER. Nous proposons ensuite notre approche du problème, la construction de cartes de *difficulté* (*difmap*), de la version brute au modèle probabiliste d'observation. Enfin, nous verrons d'autres types de données utilisables, sous la forme de cartes classifiées ou de données additionnelles provenant d'autres sources.

4.1 Etat de l'art

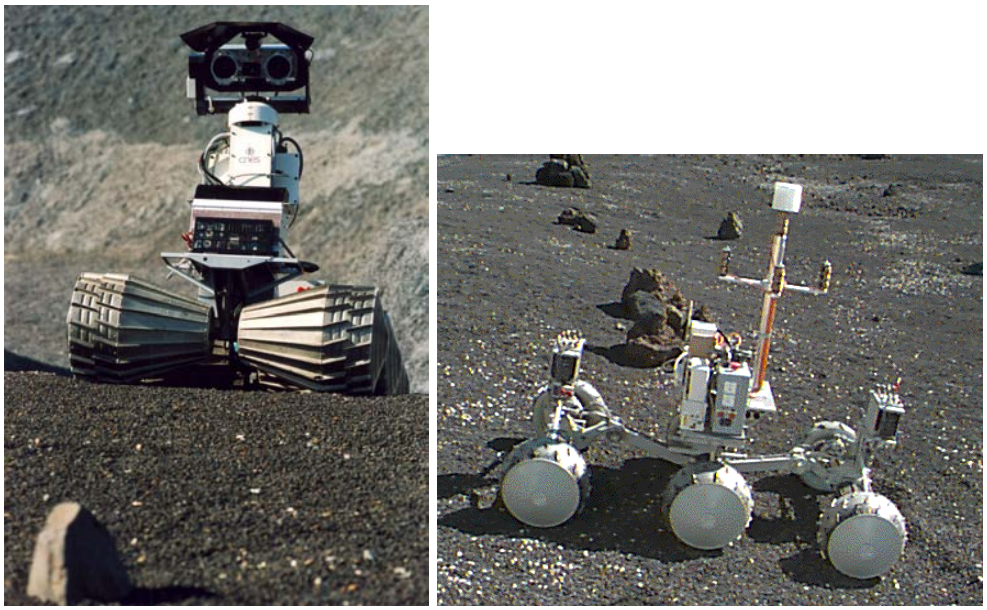
Nous proposons de présenter ici deux méhodes de perception dite "par simulation d'action" proches de celle que nous avons développée et que nous exploitons pour l'identification du *contexte*, puis d'évoquer quelques autres méthodes notables de la littérature.

4.1.1 Perception du terrain "par simulation d'action"

La méthode CNES

L'équipe de robotique du CNES a proposé une méthode de construction de *carte de navigation* (ou carte de traversabilité) intéressante, mise au point pour les robots Eve puis Iares (figure 4.1) [Delpech et al., 1998]. La construction d'une telle carte repose tout d'abord sur la construction

d'un MNT (carte des élévations $z = f(x, y)$, voir 1.3.1 et 3.2.2) grâce aux données de stéréo-vision acquises à bord du robot. Pour effectuer la partie simulation d'action, le robot est représenté par un modèle simplifié constitué d'un essieu unique muni de deux roues d'empreinte carrée. Deux paramètres servant par la suite à classifier le terrain sont calculés pour chaque cellule du MNT : une discontinuité de terrain maximum ($z_{max} - z_{min}$) sous l'empreinte de la roue et la pente maximum de l'essieu lorsqu'il est positionné sur les z_{max} du terrain, avec différentes orientations (le cas le pire est retenu, pour une stratégie conservatrice). La classification est alors réalisée en utilisant des seuils et en introduisant un hystérésis. Si la pente est au-dessous d'un seuil δ_1, la zone (cellule de la carte de navigation) est considérée comme *exécutable* ; entre δ_1 et δ_2 elle est étiquetée *exécutable mais non planifiable* (i.e. la zone est considérée comme franchissable par le robot mais on ne pourra pas planifier de chemin passant par cette zone) ; et enfin au-dessus de δ_2 la cellule est simplement *non exécutable*. Une des particularités de cette méthode repose dans la méthode de *fusion*. En effet, cette dernière n'est pas réalisée au niveau du MNT pour éviter le problème des "marches d'escaliers" qui apparaissent bien souvent suite à des erreurs de localisation et plus particulièrement d'une mauvaise estimation de l'élévation et/ou de l'attitude du robot dans sa position d'acquisition des données qui ont alimenté ce MNT. Ainsi, pour les zones en chevauchement, priorité est donnée aux perceptions les plus récentes. D'autre part, pour prendre en compte la dérive des systèmes de localisation et de locomotion, une dilatation des régions interdites et non planifiables est réalisée (en d'autres termes, il s'agit d'une augmentation de la taille des zones obstacles).



(a) Eve (b) Iares

FIG. 4.1 – Les rovers Eve et Iares du CNES

La méthode JPL-NASA pour le programme MER

La méthode développée par le JPL de la NASA pour les rovers du programme MER (figure 4.2), nommée GESTALT (*Grid-based Estimation of Surface Traversability Applied to Local Terrain* [Golberg et al., 2002]), est une évolution de l'algorithme *Morphin* mis au point à l'origine

par le Robotics Institute de CMU [Singh et al., 2000] Cette méthode se base sur des mesures continues de traversabilité et effectue elle aussi une fusion des *cartes de traversabilité* plutôt que de la représentation directe du terrain, pour essayer de réduire les effets des erreurs d'estimations de position. Les cartes locales fusionnées sont des cartes de "*goodness*" dans l'espace de configuration (figure 4.3).

FIG. 4.2 – L'un des deux rovers du programme MER

Pour chaque cellule du terrain, les angles de roulis/tangage et une grandeur de *roughness* du terrain sont estimés en utilisant une méthode des moindres carrés pour positionner un plan sur les points 3D issus de la stéréo-vision et positionnés dans la grille dans un *patch* de la taille du robot. La grandeur de *roughness* est le résidu χ^2 du placement du plan. La valeur de *goodness* est alors le minimum de ces trois grandeurs, préalablement normalisées.
Une grandeur de *certitude* est également associée à chaque cellule de la carte de traversabilité. Il s'agit d'une valeur dans [0,1] qui est fonction du nombre de points dans le *patch* et de leur distribution.
Les cartes de *goodness* sont fusionnées en effectuant une moyenne pondérée des différentes valeurs de *goodness*, les poids utilisés étant proportionnels aux valeurs de *certitude*. En effet, lors des fusions de cartes locales, une pondération par "l'âge" est réalisée en multipliant la valeur de *certitude* par une valeur inférieure à l'unité et proportionnelle à la distance parcourue par le rover depuis la dernière mise à jour de la carte. Ainsi, comme pour l'exemple du CNES, les données les plus récentes sont préférées, même si celles plus anciennes gardent une influence. Finalement la valeur de *traversabilité* est définie comme le produit (*goodness* * *certainty*).

4.1.2 Autres méthodes

D'autres méthodes intéressantes effectuent une classification de terrain pour la navigation de robot en extérieur. [Seraji et al., 2001b, Seraji et al., 2001a] utilisent un formalisme de logique floue, exploitant une définition qualitative "perceptuelle" "linguistique" de la traversabilité du terrain, par opposition aux définitions mathématiques (utilisant une fonction analytique de caractéristiques géographiques telles que la pente par exemple). L'approche logique floue a l'avantage d'être intuitive et facilement compréhensible, mais certaines personnes lui reprochent aussi parfois un côté un peu "bricolage".

FIG. 4.3 – Un exemple de grille d'occupation locale générée par GESTALT, avec les obstacles en foncé, d'après [Golberg et al., 2002].

[Castelnovi et al., 2005] effectuent un contrôle réactif de la vitesse d'un engin à roue sur la base d'une détection de *roughness* de terrain, réalisée à l'aide d'un capteur laser 2D incliné balayant le terrain se trouvant devant le véhicule. Une analyse des hauteurs des points du sol perçus est réalisée, les comparant avec les références de sol plat enregistrées à l'initialisation du système. Des attributs sont générés à partir de ces données (ils sont au nombre de quatre : la variance de la distribution des élévations du profil de terrain, la moyenne de la pente absolue, sa moyenne quadratique et la moyenne quadratique calculée avec une moyenne nulle de surface plate idéale). L'analyse de ces attributs consiste alors en leur comparaison avec des seuils prédéfinis. En fonction de cette analyse, le système décide si la vitesse doit être augmentée, diminuée ou maintenue à son niveau actuel.

4.2 Cartes de "difficulté" (*difmap*)

Nous proposons une représentation des données terrain proche des méthodes de *perception par simulation d'action* qui viennent d'être évoquées. La principale différence vient du fait que la notion de traversabilité est calculée à partir du placement d'un modèle complet de la structure du robot (même s'il est discrétisé, et donc légèrement approximé). Nous nommons les cartes de traversabilité obtenues par cette méthode *cartes de difficulté* (*difmap*).

4.2.1 Construction d'une *difmap* "brute"

Une *difmap* est avant tout une grille cartésienne d'échelle au moins égale à celle du MNT dont on se sert pour la construire. Cette échelle peut être directement choisie comme un multiple de celle du MNT, ou déterminée en fonction de la dimension du robot. Une *difmap* se construit de la manière suivante : en chaque cellule de la grille prédéfinie, on applique pour différentes orientations la fonction de placement de la structure du robot, `placeRobot`, présentée en 3.2.2. L'incrément de θ, angle d'orientation du robot, est choisi de manière à garantir un recouvrement suffisant des empreintes des roues entre deux positions (orientations) successives. Ainsi, nous posons cet incrément égal à :

$$\delta\theta = \arctan(2R/e)$$

où R est le rayon de la roue et e la longueur de l'essieu (figure 4.4).

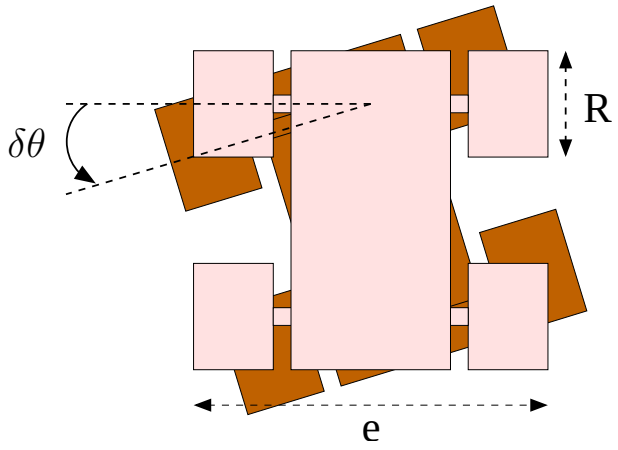


FIG. 4.4 – Vue de dessus d'un robot tel que Dala. Incrément d'angle $\delta\theta$ entre deux orientations successives du robot utilisées pour effectuer le placement, en vue de la construction d'une *difmap*.

Comme nous l'avons vu auparavant, l'opération de placement de robot nous fournit des prédictions des angles de configuration et d'attitude. Pour chaque placement, une valeur normalisée de *difficulté* peut être calculée d'une manière similaire à ce qui est réalisé pour la navigation de type *RoughNav* (3.2.2) : la difficulté $diff$ attribuée à un placement est le maximum des rapports entre chaque angle prédit (ou garde au sol prédite, le cas échéant) et son maximum autorisé du fait des contraintes sur la structure du robot (généralement fourni par l'opérateur) :

$$diff = max_i(angle_i/angle_{max_i}, 1 - (gas/gas_{max})) \tag{4.1}$$

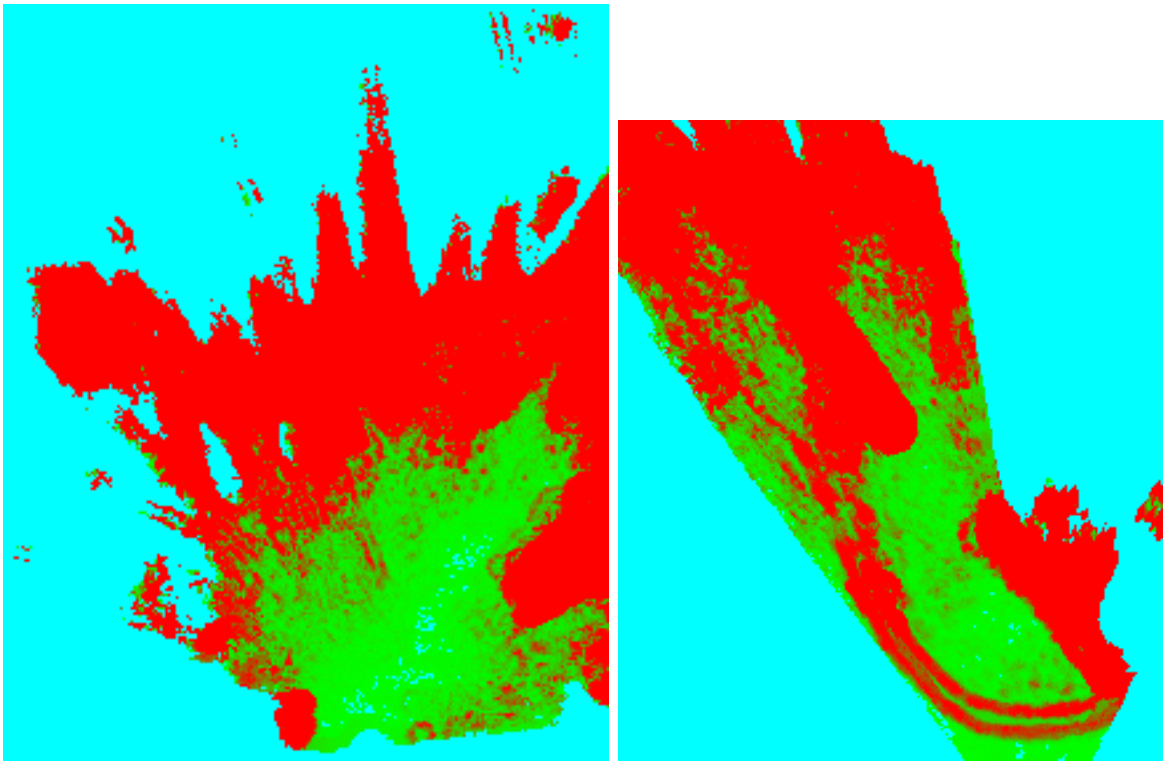
où gas désigne la garde au sol prédite (distance minimum existant entre le terrain et le bas de caisse du robot) et gas_{max} la garde au sol maximale de l'engin, lorsqu'il est sur terrain plat (i.e. sur terrain parfaitement plat, $gas = gas_{max}$). Une configuration considérée comme non acceptable (un des paramètres prédit dépasse le maximum autorisé, ou le placement n'est pas réalisable) se verra affecter une *difficulté* de 1.

La difficulté affectée à une cellule est enfin le maximum des difficultés obtenues pour les différentes orientations testées.

La figure 4.5 montre deux exemples de *difmap* "brutes".

4.2.2 Fusion

A bord du robot, une *difmap* globale ayant les mêmes dimensions géométriques que le MNT global est régulièrement mise à jour en fusionnant des *difmaps* locales. La stratégie simple adoptée, pour le moment, pour la fusion d'une *difmap* locale dans la globale est le remplacement des anciennes données par les nouvelles là où il y en a. Sont toutefois exclues de la procédure de placement les bandes sur les bords de la grille de la carte locale, d'une largeur égale au maximum entre demi-largeur et demi-longueur du robot, car on n'y dispose pas d'assez d'informations sur le terrain pour effectuer le placement du robot complet.



(a) *difmap* de taille 22x30 mètres carrés

(b) *difmap* de taille 15,5x13,5 mètres carrés, présentant un trottoir courbe

FIG. 4.5 – Deux exemples de *difmap* brutes, représentées en dégradé de couleur vert-rouge : plus la cellule est rouge, plus la difficulté est grande, à l'exception de la zone extérieure tout autour en bleu qui est inconnue (le MNT ne contient pas d'information dans cette zone permettant de compléter la *difmap*).

4.2.3 Modèle probabiliste d'observation

L'objectif de la carte qualitative de représentation du terrain est d'identifier 3 classes de terrain, qui sont les types de contexte correspondant aux modes de navigation envisagés dans notre application (voir 3.4.2) :

1. **terrain plat**,
2. **terrain accidenté** franchissable par le robot,
3. **obstacle** (i.e. terrain non franchissable).

La donnée de *difficulté* qui vient d'être introduite est l'attribut qui va nous permettre de classifier les cellules de la *difmap*. Pour cela, nous avons besoin de distributions de probabilités des classes.

Ces classes peuvent justement être caractérisées par la difficulté. En effet, si l'on raisonne sur des données certaines, c'est à dire non entachées d'incertitudes :

- lorsque le terrain est *plat*, les angles d'attitude sont nuls et la garde au sol (le cas échéant) est maximale, ce qui correspond à une *difficulté nulle* ;
- lorsqu'il est *accidenté mais franchissable*, l'attitude ou la configuration est non nulle mais reste dans les limites admissibles et la garde au sol est strictement positive, ce qui correspond à une *difficulté* $diff \in]0;1[$;

– enfin, le terrain n'est *pas franchissable* (*obstacle*) lorsqu'au moins l'un des angles d'attitude ou de configuration dépasse les limites admissibles ou bien la garde au sol prédite est négative ou nulle, ce qui implique $diff = 1$.

La difficulté étant calculée à partir de données incertaines, elle a elle-même une incertitude, que nous pouvons estimer. Considérons le cas où nous n'exploitons que les deux angles d'attitude ϕ et ψ et la garde au sol *gas* pour générer la difficulté (cas de Dala par exemple). Chacune de ces données est représentée par une moyenne et une variance, il en sera donc de même pour la difficulté, qui sera représentée par la moyenne (voir équation 4.1) :

$$diff = max\left(\frac{|\phi|}{\phi_{max}}, \frac{|\psi|}{\psi_{max}}, 1 - \frac{gas}{gas_{max}}\right)$$

et la variance :

$$\sigma_{diff}{}^2 = max\left(\sigma^2_{|\phi|/\phi_{max}}, \sigma^2_{|\psi|/\psi_{max}}, \sigma^2_{1-gas/gas_{max}}\right)$$

or :

$$\sigma^2_{|\phi|/\phi_{max}} = \frac{\sigma_{|\phi|}{}^2}{\phi_{max}{}^2},$$

le même raisonnement étant valable aussi pour ψ.
D'autre part,

$$\sigma^2_{1-gas/gas_{max}} = \frac{\sigma_{gas}{}^2}{gas_{max}{}^2}$$

En utilisant cette représentation et les caractérisations des trois classes que nous avons formulées en terme de difficulté, nous pouvons alors modéliser la densité de probabilité des classes *Terrain Plat* et *Obstacle* par des répartitions gaussiennes de la difficulté, de moyennes respectivement 0 et 1, et d'écart-type σ_{diff}. La densité de probabilité de la classe intermédiaire, à savoir *Terrain Accidenté*, est modélisée par le complémentaire des densités des deux autres classes : $p(diff|Accidente) = 1 - (p(diff|Plat) + p(diff|Obstacle))$. Ces trois densités de probabilités $p(diff|Plat)$, $p(diff|Accidente)$ et $p(diff|Obstacle)$ sont illustrées dans la figure 4.6.

Utilisation dans le système multi-modes de navigation

Ces densités de probabilité constituent le *modèle d'observation* (voir section 2.3.4) du système multi-modes de navigation introduit dans le précédent chapitre, en 3.4.2. En effet, dans ce système, les contextes dédiés aux trois modes disponibles *FlatNav* (m_1), *RoughNav* (m_2) et le "non-mode" *Stop* (m_2) sont respectivement *Terrain Plat*, *Terrain Accidenté* et *Obstacle*. Ainsi, dans le système de sélection de mode, l'observation notée O_t dans le chapitre 2 est ici la difficulté $diff$ attribuée à la cellule de la *difmap* sur laquelle est positionné le robot à l'instant t, et les trois probabilités d'observation $p(diff_t|x_i)$ avec $i \in \{1, 2, 3\}$ sont calculées à partir des trois densités illustrées dans la figure 4.6.

4.2.4 Découpage en régions

Dans nos expérimentations, la *difmap* d'origine est constituée de cellules de la taille de celles du MNT, soit en pratique 5 à 10 cm de côté, et sur chaque cellule une difficulté est évaluée. Afin de pouvoir manipuler cette carte pour la navigation, il nous faut la découper en régions plus grandes. Pour cela, une re-discrétisation de la *difmap* est réalisée, d'une taille paramétrable et définie en nombre de cellules de la *difmap* d'origine, sur laquelle les difficultés ont été affectées.

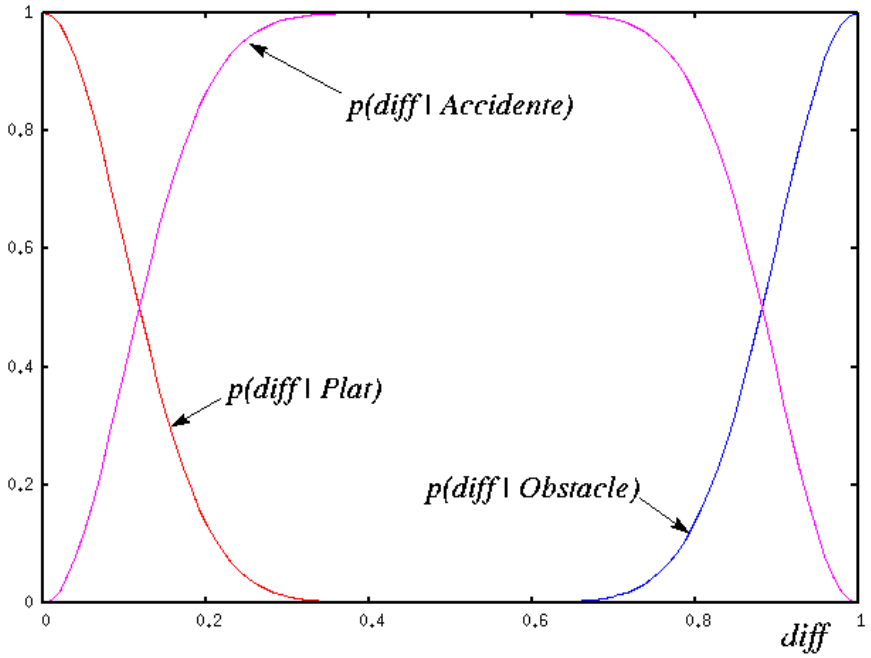


FIG. 4.6 – *Modèle d'observation* pour le HMM de sélection des trois modes de navigation. Densités de probabilité de la difficulté suivant la classe de terrain : $p(diff|Classe)$

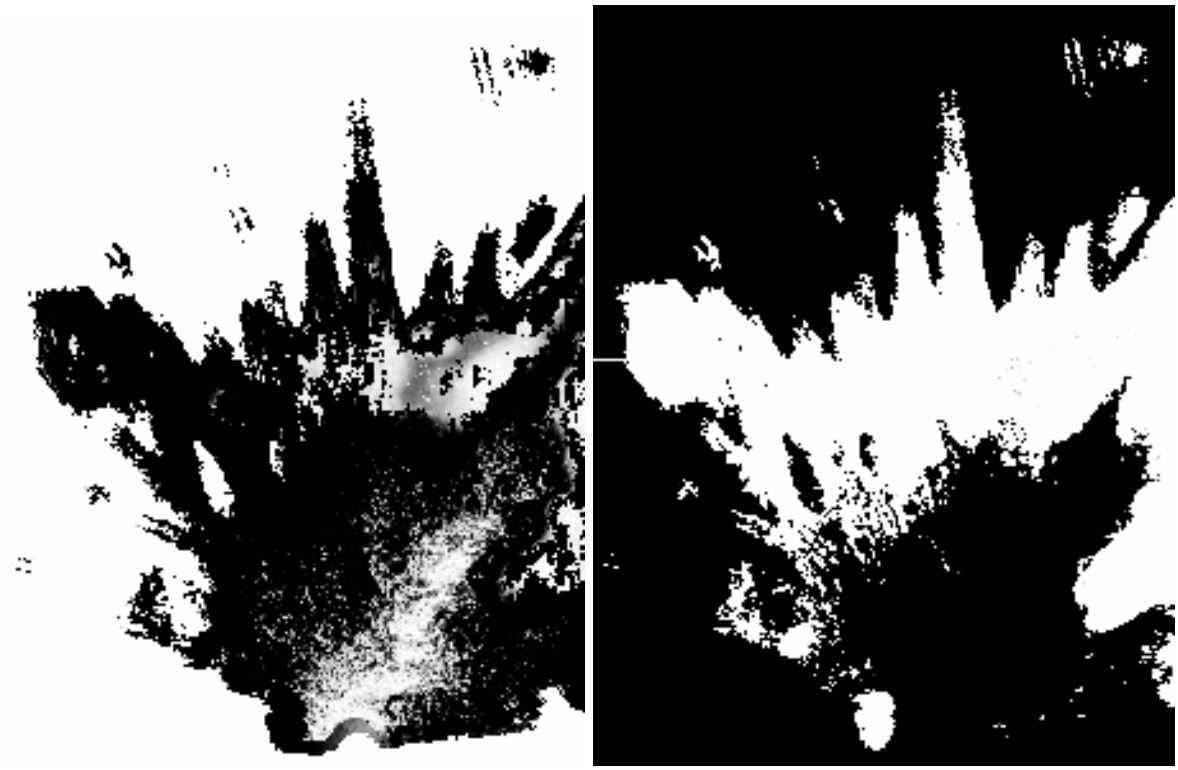
La difficulté affectée à une macro-cellule de cette nouvelle *difmap* de taille plus réduite, mais couvrant la même zone de terrain, est la moyenne des difficultés contenues dans cette macro-cellule (il y en a autant que de cellules dans la *difmap* d'origine). On peut alors appliquer les mêmes fonctions de densité de probabilité et calculer les probabilités partielles de contexte pour cette nouvelle *difmap*.

4.3 Carte classifiée

Nous pouvons également exploiter une autre méthode de construction de modèle qualitatif pour représenter le terrain, reposant sur une procédure de classification bayesienne.

4.3.1 Classification sur attributs géométriques

Pour donner au rover des informations sur la *traversabilité* du terrain perçu, le LAAS a développé une méthode qui produit une description du terrain en termes de *classes de navigabilité* sur la base des données fournies par la stéréo-vision [Lacroix et al., 2002]. Cette méthode est une procédure de classification qui produit une carte constituée de polygones étiquetés par une probabilité. La zone perçue est d'abord discrétisée en cellules, en "respectant" les caractéristiques du capteur. En effet, la résolution des cellules diminue avec la distance, de même que la résolution des données stéréo perçues. Chaque cellule est ensuite étiquetée par un classificateur Bayesien non paramétrique, utilisant une base d'apprentissage construite hors ligne par un opérateur à partir d'images "tests" (*apprentissage supervisé*). Puis, en ligne, un vecteur d'attributs de dimension 10 est calculé pour chaque cellule (ces attributs étant : densité de points 3D, variance sur l'altitude, vecteur normal moyen et variances correspondantes). La formule de Bayes permet



(a) $p(diff|Plat)$ (b) $p(diff|Obstacle)$

FIG. 4.7 – La *difmap* vue figure 4.5, après "probabilisation". La figure de gauche représente la probabilité partielle de $diff$ sachant *Terrain Plat* en niveaux de gris : plus la probabilité est proche de 1, plus la cellule correspondante est blanche (toujours à l'exception de la zone inconnue tout autour). On peut remarquer ici une sorte de vallée de terrain plat, chemin qu'a justement emprunté le robot.

alors la détermination des probabilités partielles qu'une cellule donnée corresponde à une classe de traversabilité prédéfinie. La figure 4.8 montre un résultat de classification en considérant deux classes de terrain (*plat* et *obstacle*). On peut bien entendu effectuer avec cette même méthode une classification en trois classes de terrain (*plat*, *accidenté* et *obstacle*) pour nos besoins concernant les modes de navigation.

4.3.2 Classification sur attributs de texture

On peut ajouter à cette classification "géométrique" un classificateur Bayesien de la nature physique du terrain, à partir d'attributs de texture et/ou de couleur. Ceci peut permettre en particulier de disposer d'informations de contexte utiles pour le choix du mode de locomotion, en particulier concernant la nature cohésive du sol (cf. 3.4.1 et annexe B). Dans ce cadre, des études ont été réalisées dans le projet R2M. Il s'agit d'effectuer une analyse de la texture du sol afin de déterminer sa nature [Martinet et al., 2006]. Des attributs de texture sont calculés à partir d'une image couleur d'une des deux caméras du banc de stéréo-vision du robot, et comparés de manière statistique avec des données d'une banque d'images apprises dans les mêmes conditions

FIG. 4.8 – Un exemple de résultats de la classification Bayesienne par attributs géométriques. De gauche à droite : image montrant les pixels auquels l'algorithme de stéréovision a pu associer une profondeur, description de la zone perçue avec probabilités partielles pour chaque cellule d'être un obstacle (plus c'est clair plus c'est traversable), et re-projection de ces probabilités dans le repère capteur.

de perspective (l'influence d'une forte perspective est en effet la principale difficulté à affronter pour l'étude de la texture). Ces travaux n'ont malheureusement pas encore été menés jusqu'à l'intégration complète sur rover. Nous ne disposons donc pas encore de données de contexte générées automatiquement pour la sélection de modes de locomotion.

4.4 Données additionnelles

D'autres types de données peuvent contribuer à la description du contexte, voire se subsistuer à la représentation par *difmap*. Nous proposons deux solutions envisagées dans le cadre de ces travaux.

4.4.1 Carte de traversabilité aérienne

Nous pensons que les données sur le terrain obtenues grâce à la stéréo-vision à bord du robot terrestre peuvent être avantageusement combinées avec des données issues de la perception d'un engin aérien, tel qu'un dirigeable, survolant la zone dans laquelle l'engin terrestre évolue. Une telle collaboration entre l'aérien et le terrestre pour la perception est actuellement en cours d'étude, le principe étant d'utiliser les travaux de [Bosch et al., 2006] avec le dirigeable Karma du LAAS-CNRS (figure 4.9) pour obtenir une carte probabiliste des zones planes de l'environnement survolé, la grille cartésienne construite par Karma contenant des probabilités partielles que le terrain soit globalement plan ou non.

4.4.2 Données "initiales" fournies par un opérateur

Comme le disent les auteurs de [Biesiadecki et al., 2005], "les humains sont très bons en analyse de terrain pour la planification de mouvement" ou pour sa classification. Les données de contexte peuvent donc également être fournies par un opérateur humain étudiant les images fournies par le robot lui-même ou par un autre engin survolant le terrain que le robot doit parcourir. L'analyse du terrain est ainsi la plupart du temps réalisée de cette manière pour les Mars Exploration Rovers, à partir de "l'imagerie orbitale" (fournie par le module restant en orbite autour de Mars) et des images fournies par les caméras de navigation des rovers (l'utilisation de cette analyse débouche toutefois le plus souvent sur une planification de chemin également réalisée par les humains, pour déterminer le parcours du robot pendant la journée à venir).

FIG. 4.9 – Le ballon dirigeable du LAAS-CNRS : Karma

Les humains peuvent alors fournir une pseudo-probabilité partielle de chaque classe pour des régions de la carte, spécifiées par eux-mêmes, ou bien leur attribuer une grandeur de *difficulté* du terrain, d'après eux, qui sera ensuite transcrite en des probabilités partielles de classe suivant les distributions précisées dans la section 4.2.3.

Si de telles données initiales sont disponibles, tout en ayant la possibilité que le robot construise sa *difmap* en ligne, on peut effectuer une combinaison de ces deux types de données afin d'enrichir le modèle.

Si le robot ne dispose pas de la capacité de générer des données d'observation nécessaires pour alimenter le système de sélection de modes, cette solution reste bien entendu la seule disponible si l'on veut profiter de données de contexte. Ce sera par exemple le cas pour le système multi-modes de locomotion pour Lama proposé en 3.4.1, car nous ne disposons pas encore en pratique de moyens d'étudier la texture du terrain afin de déterminer si le sol est cohésif ou non.

4.5 Stratégie de navigation à long terme

Les méthodes de navigation présentées dans le chapitre 3 sont des méthodes plutôt *réactives* ou à planification *locale* qui permettent donc d'atteindre de manière autonome des buts *locaux*. Par conséquent, lorsque le but à atteindre par le robot est éloigné (quelques dizaines de mètres représentent déjà beaucoup pour un rover autonome), il convient d'appliquer une stratégie de navigation plus *globale* capable de générer une liste de buts locaux qui sont transmis aux méthodes de navigation locales, afin d'éviter les pièges des impasses dans lesquels une méthode purement locale ne pourra éviter de tomber.

La stratégie que nous utilisons est issue de [Gancet et al., 2003].

4.6 Conclusion

Nous avons vu dans ce chapitre les types de représentation du terrain que nous pouvons utiliser pour fournir au système de sélection de modes les données de *contexte* dont il a besoin.

Ces représentations doivent être capables de distinguer les classes de terrain auxquelles sont dédiés les modes de déplacement.

Ainsi, les *cartes de difficulté* que nous avons développées permettent d'obtenir ce modèle d'observation pour la sélection de modes *navigation*, en prenant en compte explicitement la nature du robot et ses capacités de franchissement. Un tel modèle peut également être obtenu en utilisant une représentation qualitative du terrain construite par classification bayesienne sur la base d'attributs géométriques. L'utilisation d'attributs de texture dans le cadre d'une méthode semblable peut permettre quant à elle de fournir les données d'observation de contexte pour la sélection du mode de *locomotion*.

D'autre part, nous espérons beaucoup de l'utilisation d'images aériennes acquises par un UAV (Unmanned Aerial Vehicle) du type ballon dirigeable ou hélicoptère, et de la combinaison des données issues de leur analyse avec les données acquises au sol. Des travaux dans ce sens sont en cours dans notre laboratoire pour la réalisation de missions coordonnées entre un robot terrestre et un UAV. Ils font partie des perspectives à court terme de cette thèse.

Chapitre 5

Les moniteurs de mode

Nous avons vu dans le chapitre 2 qu'additionnellement à une observation du contexte notre système a besoin d'une évaluation (de type probabiliste) du comportement du mode de déplacement actif afin d'estimer le mode à appliquer. Cette évaluation est réalisée par ce que nous appelons des *moniteurs* de mode. En pratique, un moniteur donné se chargera de surveiller le comportement de fonctionnalités de navigation ou de locomotion.
Certains moniteurs peuvent être spécifiques à un mode, lorsqu'ils exploitent des signatures spéciales qui n'ont d'existence ou de sens que pour ce mode. Ces moniteurs ne sont alors utilisables que pour surveiller ce mode exclusivement. D'autres moniteurs sont au contraire plus génériques et peuvent s'appliquer à la surveillance de plusieurs modes ayant des caractéristiques communes.

Après une introduction visant à décrire de manière générale le principe de fonctionnement de ces moniteurs, nous présentons les différents moniteurs que nous avons développés pour évaluer le comportement des modes décrits dans le chapitre 3 et alimenter le modèle de transitions du HMM d'estimation de mode de déplacement. Nous verrons pour chacun d'eux son principe, le ou les mode(s) d'application, son fonctionnement pratique et enfin quelques illustrations de son fonctionnement.

Nous pourrons voir ensuite dans le prochain chapitre l'utilisation en pratique de ces moniteurs : comment ils peuvent être combinés puis comment ils sont exploités dans les systèmes multi-modes qui ont été proposés en à la fin du chapitre 3.

5.1 Principe des moniteurs

Un moniteur de mode évalue le comportement de ce dernier lorsqu'il est actif. Il est basé sur l'étude de signatures générées par redondance analytique (cf. section 2.1.1) ou redondance par modèle. Nous définissons ainsi pour chaque mode (ou pour un ensemble de modes ayant des caractéristiques communes) un *modèle de fonctionnement nominal*, autrement dit fonctionnement *attendu* vu notre connaissance du mode. Les signatures sont générées à partir de la comparaison entre ce fonctionnement nominal et des observations (et/ou des déductions issues de ces observations). L'étude des signatures doit permettre de déterminer une probabilité que le comportement du mode soit jugé bon, simplement acceptable ou mauvais.

Modèle de fonctionnement nominal

Nous posons les caractéristiques d'un fonctionnement dit *nominal* pour chacun des modes introduits dans le chapitre 3. Il s'agit de modèles *partiels* dans la mesure où ils décrivent simplement les caractéristiques que nous aurons les moyens de surveiller grâce à des moniteurs. Nous

ne chercherons donc pas à réaliser une synthèse *exhaustive* de ces caractéristiques mais à exhiber celles que nous avons les moyens d'observer ou d'estimer.

Objectifs pour chaque moniteur

Il s'agit de générer un indicateur par moniteur, sous la forme d'une donnée normalisée qui soit une probabilité ou pseudo-probabilité de mauvais comportement, en raisonnant uniquement sur les données considérées. Les données doivent être mises en forme d'une manière similaire pour chaque moniteur, afin de pouvoir les combiner ensuite.

5.2 Moniteur d'efficacité de la locomotion (*LEM*)

Dans certaines situations sur terrain accidenté, il arrive qu'un robot mobile ait d'importantes difficultés à avancer à cause de glissements significatifs, alors même que ses roues continuent à tourner, pour certaines, à la vitesse de rotation demandée. Nous nommons de telles situations des *fautes de locomotion*. Être capable d'évaluer la qualité de la locomotion d'un rover, et en particulier de détecter de telles fautes, est bien entendu crucial pour un rover autonome. Ceci est réalisé grâce à l'étude de ce que nous nommons des *indicateurs de cohérence de vitesses*, signatures obtenues par redondance analytique et illustrant l'efficacité de la locomotion réelle du robot. Ces indicateurs sont utilisés en ligne sur le robot pour fournir des informations à une chaîne de Markov qui fournit des probabilités partielles que le robot soit dans une situation de glissement ou de *faute de locomotion* [Peynot et al., 2003].

Le moniteur *LEM* (*Locomotion Efficiency Monitor*) concerne donc la surveillance de la locomotion par *Roulement*. Le fonctionnement nominal de ce mode de locomotion est caractérisé par la nullité des signatures (les indicateurs), c'est à dire :

- la cohérence des vitesses linéaires des roues de chaque côté du robot, deux à deux,
- la cohérence des estimations des vitesses de rotation.

Nous nommons le présent moniteur *LEM*, pour *Locomotion Efficiency Monitoring*. Pour le présenter, nous reprenons les notations de variables et de repères introduites dans la section 3.3.1 lors de la présentation de la "commande adaptée terrain" ($TALC$) dont ce moniteur est issu.

5.2.1 Indicateurs de cohérence de vitesses

Cohérence de vitesses linéaires

En tant qu'éléments du même solide, les expressions des vitesses linéaires de deux roues du robot (A et B), du même côté, doivent respecter certaines relations cinématiques entre elles à chaque instant. En effet, si K est la roue considérée (i.e. A ou B) :

$$\vec{V}_{K/\Re_G} = \vec{V}_{K/\Re_c} + \vec{V}_{C_s/\Re_G} + \overrightarrow{KC_s} \wedge \vec{\Omega}_{\Re_c/\Re_G}$$

Par conséquent, si on pose $\overrightarrow{W}(K) = \vec{V}_{K/\Re_G} - \vec{V}_{K/\Re_c}$, la relation suivante doit être vraie à chaque instant :

$$\overrightarrow{W}(B) - \overrightarrow{W}(A) \;=\; \overrightarrow{BA} \wedge \vec{\Omega}_{\Re_c/\Re_G} \tag{5.1}$$

Pour détecter les glissements, nous surveillons la composante en X_c (direction d'avancement du robot) de l'expression (5.1). $W(A)_{X_c}$ et $W(B)_{X_c}$ sont ainsi régulièrement calculés à partir des vitesses de roues mesurées ($V_{roue} = R\,\omega_{roue}$, sous l'hypothèse de roulement sans glissement)

et les estimations d'angles de contact produits par le *TALC*. Les vitesses sont considérées comme *cohérentes* si la relation (5.1) est vraie. Dans le cas contraire, la différence

$$\Delta V_{x(AB)} = \left(\vec{W}(B) - \vec{W}(A) - \overrightarrow{BA} \wedge \vec{\Omega}_{\Re_c/\Re_G}\right)_{X_c}$$

donne une estimation de l'*incohérence* entre les mesures des vitesses. La moyenne des ΔV_x calculés pour chaque côté (trois pour un châssis Marsokhod comme Lama) est un indicateur d'*incohérence* calculé à chaque échantillon de temps. Le présent moniteur fournit ainsi deux *indicateurs de cohérence de vitesses linéaires* (un pour chaque côté).

Cohérence des vitesses de rotation

A bord de nos robots Dala et Lama, une mesure de la vitesse de rotation est disponible grâce à la présence d'un gyromètre positionné de manière à mesurer la vitesse de rotation autour de l'axe vertical au robot. Nous notons cette vitesse ω_{Gyro}. Cette mesure est de bonne précision et peut donc servir de référence.
La grandeur correspondante peut également être estimée "par odométrie" en utilisant les mesures de vitesse de rotation, sous l'hypothèse de roulement sans glissement :

$$\omega_{Odo} = (V_r - V_l)/e$$

où V_r et V_l sont les vitesses linéaires respectivement de la roue droite et gauche d'un même essieu et e la longueur de cet essieu. L'expression de $V_s = V_r$ ou V_l est :

$$V_s = R\,\omega_s\,\cos(\delta_{roue})$$

où δ_{roue} est l'angle entre la vitesse linéaire de la roue et le plan d'évolution du robot : $(X_c Y_c)$ [Peynot et al., 2003]. Cet angle est estimé pour chaque roue par le TALC (voir section 3.3.1), au même titre que l'angle de contact de la roue avec le sol.

La moyenne de tous les $|\omega_{Odo} - \omega_{Gyro}|$ constitue le troisième *indicateur*, $\Delta\omega$, donnant une estimation de la *cohérence des vitesses de rotation* ; le comportement *idéal* étant caractérisé par : $\Delta\omega = 0$.
La figure 5.1 illustre le comportement de ces trois indicateurs sur le robot Lama, en particulier dans des situations de *fautes de locomotion*.

5.2.2 Monitoring de locomotion

Les trois *indicateurs de cohérence de vitesses* sont utilisés comme *attributs* dans une procédure de classification probabiliste : un modèle de Markov caché (HMM) évalue la situation du robot en ligne à chaque instant d'après les probabilités à l'instant précédent, les valeurs courantes des attributs et leur comparaison avec des prototypes enregistrés pendant une phase préliminaire d'apprentissage supervisé. Nous avons choisi de réaliser cet apprentissage car nous n'avons pas pu trouver une possibilité de réaliser une classification paramétrique.

Apprentissage supervisé

Cette opération vise à collecter des informations alors que le comportement effectif du robot est connu à chaque instant. De nombreuses expérimentations ont été réalisées avec le robot Lama, pour qui nous disposons de plus d'informations du fait de ses six roues indépendantes et qui a une locomotion offrant plus de possibilités qu'un *ATRV* comme Dala par exemple. Nous distinguons des situations dans trois catégories principales, correspondant à trois états du robot :

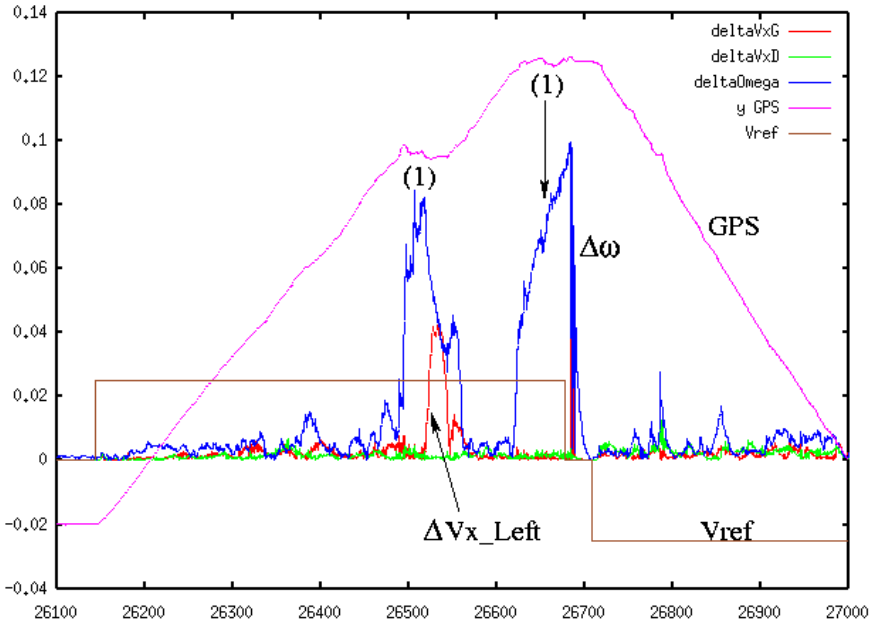


FIG. 5.1 – Les trois *indicateurs de cohérence des vitesses* (ΔVx_{Left}, ΔVx_{Right} et $\Delta\omega$) en fonction du temps, en situations de *fautes de locomotion* (1). On peut constater des augmentations significatives de ces indicateurs lors des fautes de locomotions, révélées ici par le GPS centimétrique embarqué à bord du robot pour cette expérience, et qui montre l'évolution réelle de la position du rover le long de son axe de déplacement.

- **Etat 0** (s_0) : Situation "facile" pour le rover, avec locomotion efficace. Absence de glissements *anormaux*[1].
- **Etat 1** (s_1) : Situation difficile pour le rover. Quelques glissements longitudinaux *anormaux* se produisent, mais le robot progresse toujours.
- **Etat 2** (s_2) : *Faute de locomotion*. A cause de glissements excessifs, le rover n'avance plus, même si ses roues continuent à tourner comme demandé.

Au cours des expérimentations réalisées pour l'apprentissage, l'état courant est déterminé par un opérateur observant la scène (d'où l'appellation d'apprentissage *supervisé*). Sa décision et les autres données intéressantes (les deux consignes de vitesses pour le robot V_{ref} et ω_{ref} et les trois indicateurs de cohérence de vitesses) sont enregistrées pour être ensuite utilisées en tant que prototypes dans la procédure de classification.

Nos rovers étant des plates-formes de type *skid-steering*, leur comportement est fortement dépendant du couple (V_{ref}, ω_{ref}). En conséquence, plusieurs séries d'expérimentations ont été effectuées avec différentes valeurs pour ce couple, chaque valeur correspondant à une *base de prototypes* contenant les attributs enregistrés.

D'autres opérations d'apprentissage supervisé pourraient être avantageusement réalisées avec plus de variété de conditions de terrain (e.g. terrain mouillé) pour constituer autant de bases de prototypes. Toutes ces données pourraient également être enregistrées dans des bases de prototypes communes mais notre évaluateur d'efficacité de locomotion serait alors bien moins discriminant.

[1] Nous excluons ici le cas des glissements *latéraux* que nous n'avons pas les moyens de détecter sur nos robots.

Evaluation en ligne de la situation

L'évolution dans le temps de la situation du robot doit bien entendu être prise en compte (par exemple, une faute de locomotion n'arrive généralement pas soudainement, mais à la suite de symptômes s'aggravant progressivement). C'est pourquoi une chaîne de Markov est utilisée (c'est en fait un HMM), plutôt qu'un processus de classification direct [Postaire, 1987], pour calculer à chaque instant la probabilité que le robot soit dans l'état s_0, s_1 ou s_2.

HMM

Si l'observation obs_t est réalisée au temps t, la probabilité que le robot soit dans l'état s_k est :

$$p(s_{k,t}|obs_{1:t}) \;=\; \eta\, p(obs_t|s_k) \sum_{i=0}^{K-1} p_{ik}\, p(s_{i,t-1}|obs_{1:t-1}) \tag{5.2}$$

où :

- $obs_{1:t}$ représente les observations effectuées jusqu'au temps t,
- p_{ik} est la probabilité de transition de l'état s_i à l'état s_k,
- K est le nombre d'états ($K = 3$ ici),
- $p(obs_t|s_k)$ la probabilité que l'observation obs_t soit faite sachant que le rover se trouve dans l'état s_k (issue du modèle d'observation),
- η le coefficient de normalisation permettant d'assurer que $\sum_{k=0}^{K-1} p(s_k|obs_{1:t}) \;=\; 1$.

Probabilités de transition

Les probabilités de transition de cet automate ont été posées empiriquement, suite aux multiples expérimentations réalisées dans la phase d'apprentissage supervisé. Leurs valeurs peuvent être vues sur la figure 5.2. Elles peuvent être adaptées suivant la sensibilité voulue pour la détection de fautes de locomotion ou encore le type de terrain, si le robot est capable de le déterminer.

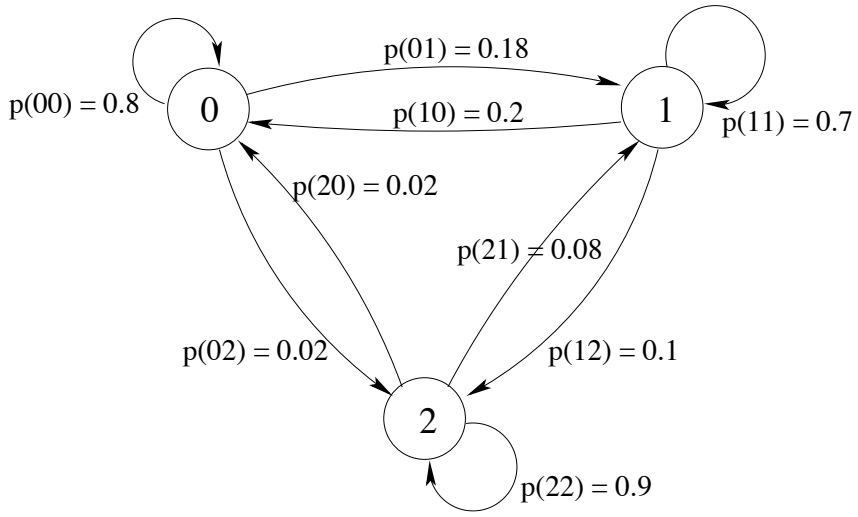


FIG. 5.2 – Etats et probabilités de transition pour le HMM du moniteur d'efficacité de locomotion.

Densité de probabilité

Pour calculer la densité de probabilité $p(obs|s_k)$, nous utilisons la méthode des *plus proches voisins* [Postaire, 1987]. Cependant, la recherche complète en ligne des plus proches voisins pour chaque échantillon serait trop coûteuse en temps de calcul. C'est pourquoi les probabilités $p(obs|s_k)$ sont calculées tout d'abord hors-ligne pour tous les échantillons d'apprentissage. Ensuite, en ligne, pour chaque nouvel échantillon le système effectue seulement une recherche du prototype le plus proche et les densités de probabilités partielles attribuées à cet échantillon sont celles qui ont été associées hors ligne à ce prototype.

Calcul en ligne des probabilités d'états

A chaque échantillon au temps t sont associées les données utiles suivantes : $V_{ref}(t)$, $\omega_{ref}(t)$ et les trois valeurs d'*attributs* à t (i.e. les trois indicateurs de cohérence de vitesse). La première opération consiste à choisir la base de prototypes la mieux adaptée à la situation courante (c'est à dire avec le couple (V_{ref}, ω_{ref}) le plus proche). Ensuite, nous recherchons le prototype le plus proche dans cette base et utilisons les probabilités d'états associées comme estimation des densités de probabilité pour cet échantillon.
Tous les éléments nécessaires étant alors disponibles, les probabilités d'états pour l'échantillon courant peuvent être calculées grâce à l'équation (5.2). Le cycle complet de calcul en ligne réalisé pour chaque échantillon est illustré par la figure 5.3.

Illustration

Les probabilités partielles d'états sont celles qui serviront pour la mise à jour du *modèle dynamique* (ou modèle de transitions) du superviseur de sélection de modes de déplacement. La figure 5.4 montre un exemple de résultats obtenus en comparaison avec l'opinion d'un opérateur humain.

Le moniteur présenté ici fournit donc une évaluation en ligne de l'efficacité de la locomotion d'un robot mobile, permettant en particulier la détection de *fautes de locomotion*. Cette méthode n'exploitant que des informations de vitesses donne des résultats satisfaisants, sous réserve toutefois que le type de terrain ne diffère pas trop des terrains sur lesquels l'apprentissage a été réalisé. Elle bénéficierait très probablement de l'utilisation de données supplémentaires (telles que le courant dans les moteurs des roues) permettant de générer des attributs additionnels.

5.2.3 Version simplifiée

Le robot Dala a une structure plus simple et plus rigide que Lama, avec seulement deux essieux rigidement liés. Nous disposons donc de moins de redondance d'informations sur les vitesses linéaires des différents éléments de la structure. De plus, en pratique, nous n'avons pas accès aux informations de vitesses des roues individuellement ni au contrôle séparé de chacune d'elle. Il nous est donc impossible de générer des indicateurs de cohérence de vitesses linéaires comme c'est le cas pour Lama. De ce fait, nous avons dû réaliser pour Dala une version simplifiée de ce moniteur de locomotion, reposant sur un seul des indicateurs de cohérence de vitesse : celui des vitesses de rotation (comparaison entre la mesure du gyromètre et l'estimation par odométrie : $\Delta\omega = \omega_{Odo} - \omega_{Gyro}$).

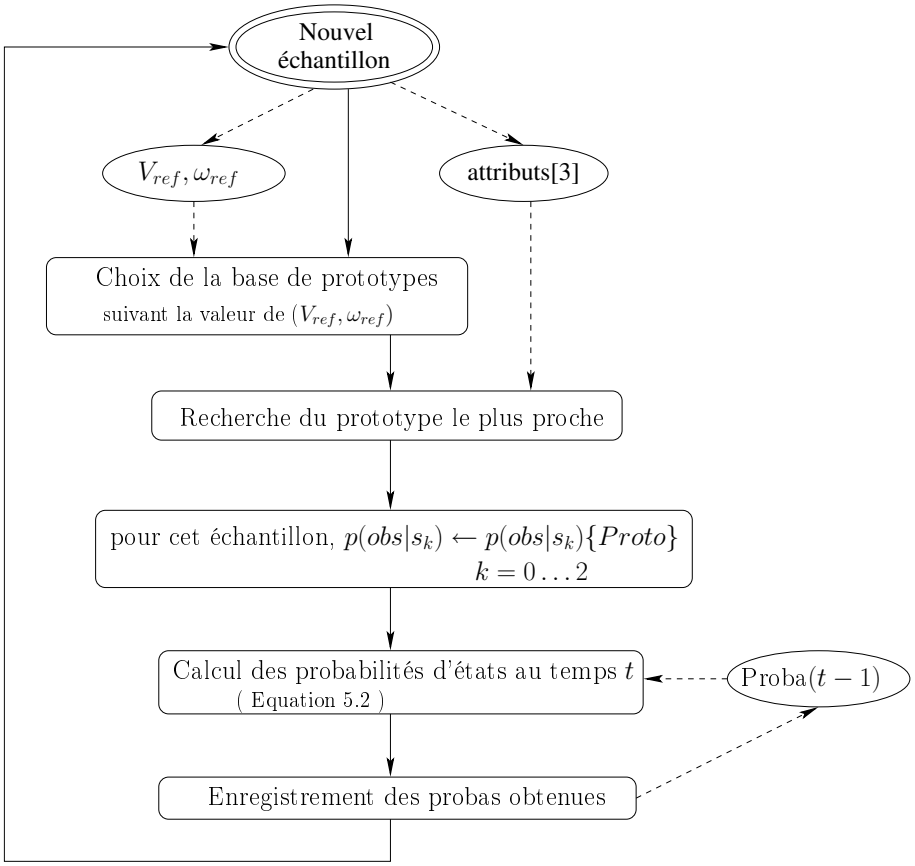


FIG. 5.3 – Principe de calcul en ligne des probabilités d'états pour le moniteur d'efficacité de locomotion.

5.3 Détection d'Accident de Terrain (*DAT*)

5.3.1 Principe

Ce moniteur est d'un type différent, car s'il exploite des données proprioceptives comme le *LEM*, c'est cette fois pour vérifier si le contexte effectivement subi par le robot est en accord avec le mode actuellement utilisé. Plus précisément, il s'agit de vérifier que le terrain ressenti par le robot est bien plat, en particulier lorsque le mode de navigation actif est un mode comme *FlatNav* qui doit fonctionner *exclusivement* sur terrain plat. Pour ce faire nous utilisons les mesures de la centrale inertielle à bord du robot (Dala dans notre cas) et plus particulièrement celles des vitesses angulaires en tangage (ω_y) et roulis (ω_x) et éventuellement l'accélération en z : $\ddot{z}$. Sur un terrain plat, les deux gyromètres sur les axes de roulis et de tangage (ainsi que l'accéléromètre en z) ne devraient mesurer que du bruit et un éventuel biais (cf. l'exemple de la figure 5.5). Au bruit capteur peuvent également s'ajouter des données d'accélération et de vitesses angulaires dues aux vibrations mécaniques à bord du robot. Ce phénomène peut être particulièrement significatif sur un robot tel que Dala lorsqu'il évolue par exemple sur une surface goudronnée, vue sa structure rigide sans amortisseurs et ses roues équipées de pneus sculptés. Nous inclurons donc les mesures de ces vibrations dans l'appellation de "bruit" de nos signaux.

Le modèle de fonctionnement nominal concernant ce moniteur dans le mode *FlatNav* est donc

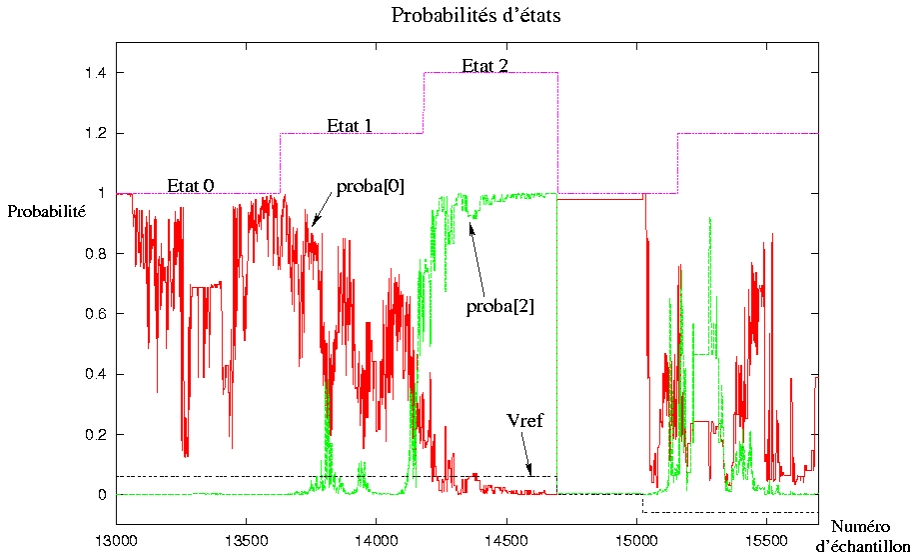


FIG. 5.4 – Comparaison entre probabilités d'états calculées et l'état courant selon l'opérateur.

que les mesures ω_x, ω_y (et $\ddot{z}$) ne contiennent que du "bruit" au sens qui vient d'être vu.

Afin d'absorber en partie le bruit des signaux (pourtant déjà filtrées lors de l'acquisition) et d'accumuler de l'évidence au cours du temps, nous raisonnons sur un niveau d'*énergie* des signaux, intégrée sur une fenêtre temporelle d'une largeur Δt au moins égale à la période à laquelle les données de ce moniteur sont prises en compte par le module principal de sélection de mode :

$$E(t) = \sum_{i=0}^{K-1} x_{t-i}^2 / K$$

où K est le nombre d'échantillons relevés dans le temps Δt, c'est à dire la partie entière de $\Delta t/\delta t$, avec δt la période d'échantillonnage des signaux de la centrale inertielle.

5.3.2 Mise en forme

De nombreux tests ont été réalisés avec le robot Dala et sa centrale inertielle pour déterminer les valeurs maximales de l'énergie des vitesses angulaires qui nous concernent, sur terrain plat, en essayant les pires conditions de vibrations (c'est à dire par exemple sur terrain goudronné, à cause des sculptures des pneus). Cela nous a permis d'identifier une valeur *MaxPlat* pour chaque donnée, maximum qui reste toutefois incertain car *toutes* les conditions que le robot peut rencontrer ne peuvent pas être testées. De même, nous pouvons déterminer des valeurs d'énergie "canoniques" pour lesquelles nous somme sûrs d'avoir affaire à un terrain accidenté : *MinAcc*. Le comportement est alors décrit par une distribution "*logit*" [Russell et al., 2003d], qui utilise une fonction *sigmoïde* pour représenter un seuil "doux", c'est à dire un seuil dont la position précise est sujette à un bruit aléatoire gaussien.

Ainsi, la pseudo-probabilité d'avoir un terrain non plat (i.e. pseudo-probabilité de *mauvais comportement*), étudiée à partir de ω_x ou ω_y, est issue du "complémentaire" de la fonction sigmoïde (figure 5.6) :

$$p(Comp|E_\omega) = 1 - \frac{1}{1 + exp(-2(MinAcc - E_\omega)/\sigma)}$$

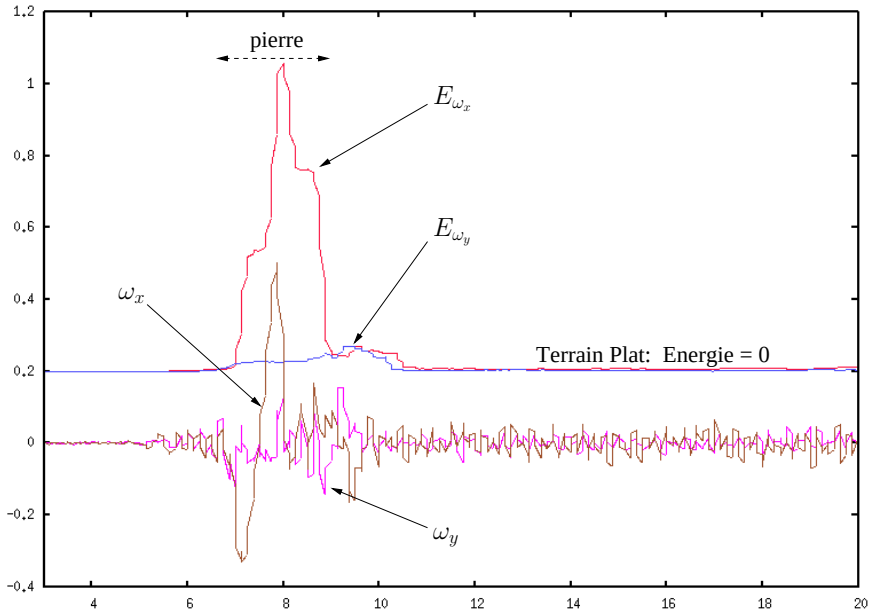


FIG. 5.5 – Vitesses angulaires ω_x et ω_y et leurs *énergies* associées alors que le robot passe sur une petite pierre, sur un terrain plat par ailleurs.

avec : $\omega = \omega_x$ ou ω_y, E_ω l'*énergie* du signal ω et $\sigma = (MinAcc - MaxPlat)/3$.

Finalement, la pseudo-probabilité d'avoir un terrain plat sous les roues du robot sera le maximum des deux valeurs ainsi obtenues pour les deux angles.

5.3.3 Illustration

La figure 5.7 montre un exemple du résultat de la mise en forme de l'énergie E_{ω_x}, lors d'une situation similaire à celle de la figure 5.5. Il est important qu'une telle situation soit nettement repérée par ce moniteur, car il s'agit d'une contradiction de l'hypothèse de contexte *terrain plat*, nécessaire pour le mode *FlatNav*.

5.4 Surveillance de l'Attitude (*SurvAtt*)

Ce moniteur s'applique au mode de navigation *RoughNav* (section 3.2.2). Il surveille la cohérence entre une observation en ligne de l'attitude et de la configuration du robot et une prédiction de ces mêmes données à partir du placement d'un modèle géométrique 3D du châssis du robot sur le MNT construit par stéréo-vision pour les besoins de ce mode de navigation. Le fonctionnement est nominal du point de vue de ce moniteur si l'observation correspond à la prédiction, c'est à dire si l'attitude et la configuration du robot déduites du MNT correspondent bien à ce qui est observé en ligne.

Ce moniteur est crucial lorsque le mode *RoughNav* est actif car ce dernier détermine par où le robot doit et peut passer précisément à partir de la fonction de placement sur le MNT (`PlaceRobot`, voir 3.2.2). Il convient donc de surveiller si cette fonction fournit des résultats cohérents avec la "réalité" du terrain.

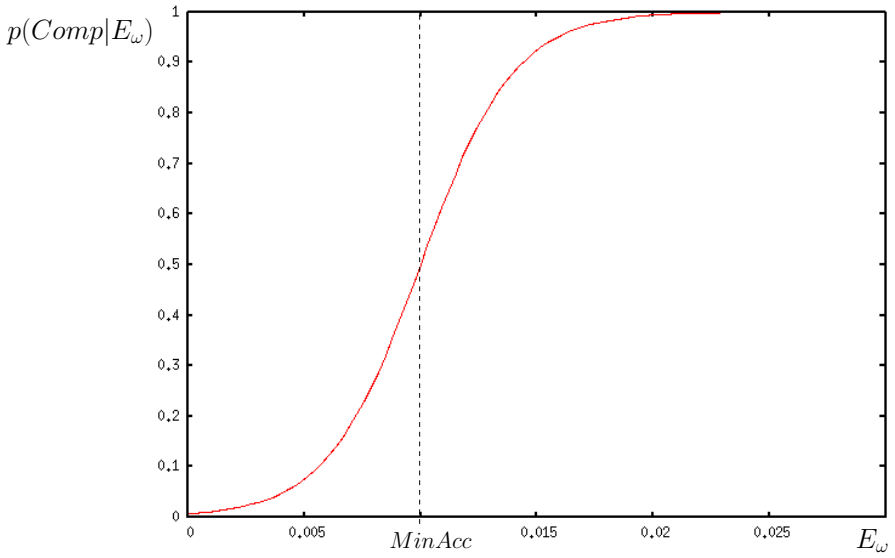


FIG. 5.6 – Allure de la pseudo-probabilité de comportement en fonction de l'*énergie* E (fonction *sigmoïde*).

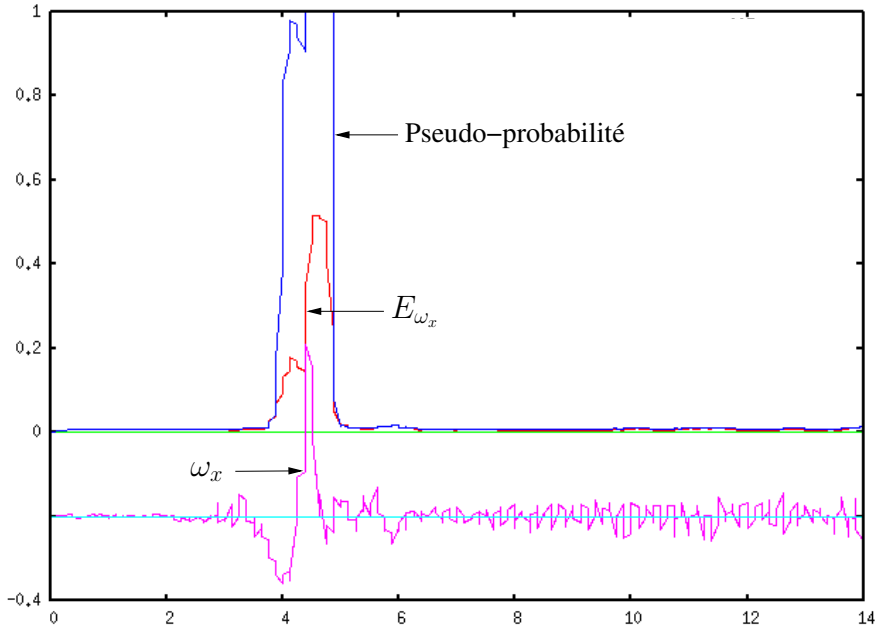
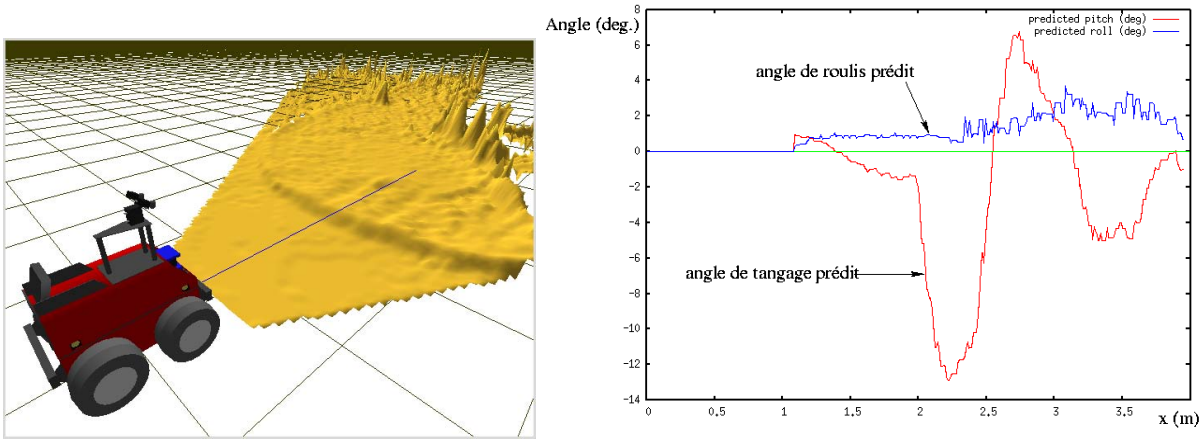


FIG. 5.7 – Pseudo-probabilité de comportement en fonction de l'énergie E (fonction *sigmoïde*), alors que le robot passe sur une petite pierre, sur un terrain plat par ailleurs.

La figure 5.8 donne ainsi un exemple de prédictions des angles d'attitude sur la trajectoire qui a été choisie par le mode *RoughNav*.



(a) Trajectoire prévue (en bleu), sur le MNT

(b) Angles d'attitude prévus sur cette trajectoire

FIG. 5.8 – Exemple de prédiction des angles d'attitude le long de la trajectoire choisie par le mode de navigation *RoughNav*, d'après la fonction de placement du robot Dala sur le MNT.

5.4.1 Sources d'erreurs pour *RoughNav*

L'erreur (connue) sur le placement du robot que nous évaluons en ligne (sous la forme d'écarts-types, voir section 3.2.2) provient essentiellement de la fusion des points du MNT. Prendre en compte uniquement ces erreurs signifierait considérer en particulier que le MNT est quasi-parfait, ce qui n'est bien entendu pas la réalité. Les principales sources d'erreurs pour le MNT sont ainsi :

- la **localisation** du robot, dont les erreurs ont pour conséquence un mauvais positionnement des points 3D et/ou une erreur d'altitude sur des points fusionnés dans les cellules du MNT dans le cas d'une erreur sur le z. En règle générale il est difficile d'évaluer ces problèmes de localisation, particulièrement à l'échelle de quelques centimètres, à moins de se servir de points de repères bien déterminés (amers) référencés globalement et dont on connaît la position précisément. C'est le cas par exemple si l'on dispose d'un récepteur GPS différentiel centimétrique à bord du robot. Cependant, il existe de nombreuses situations dans lesquelles cette technique de localisation n'est pas disponible (environnement obstrué empêchant d'être en vue d'un nombre suffisant de satellites, robot intervenant en intérieur ou sous terre dans une grotte, ou encore exploration planétaire). De plus, en général toutes les méthodes de localisation disponibles peuvent être utilisées en même temps et avantageusement fusionnées, auquel cas la meilleure méthode de localisation disponible est déjà exploitée pour fournir sa position courante au robot.
- la **calibration du banc de stéréo-vision**, et en particulier la transformation géométrique entre les deux caméras du banc, qui est la plus "sensible". Des erreurs sur ces paramètres ou sur les paramètres intrinsèques des deux caméras provoquent des problèmes d'appariement des pixels des deux images et des erreurs sur les positions des points 3D éventuellement déduits. Ce point est assez critique et les chercheurs exploitant la stéréo-vision sur des systèmes embarqués sont amenés à s'y intéresser de plus en plus. La meilleure solution semble être l'utilisation d'un système de re-calibration automatique [Lemonde, 2005].

– les **erreurs dues à la stéréo-vision** elle-même, indépendamment des erreurs importantes qui peuvent intervenir lorsque le banc de stéréo-vision n'est plus correctement calibré. Des modèles d'erreurs réalistes de la stéréo-vision ont été proposés dans la littérature, en particulier pour la réalisation de MNT [Mallet, 2001] et un de ces modèles pourrait donc être associé aux données issues des fusions successives de points dans les cellules pour donner une estimée de l'erreur plus réaliste. Cependant cela impliquerait de lourds calculs supplémentaires à réaliser en ligne. Nous préférons consacrer la puissance de calcul embarquée, déjà bien sollicitée par les seuls algorithmes de stéréo-vision dense, aux appels à la fonction de placement du robot effectués régulièrement pour la navigation dans le mode *RoughNav*.

D'autre part, le modèle de représentation du terrain est supposé *rigide*, c'est à dire non déformable, même lorsqu'un robot assez lourd se pose dessus. On peut imaginer diverses situations dans lesquelles une telle hypothèse va provoquer des écarts avec la réalité du monde lors de l'étude du posé de robot. En effet, un robot peut voir ses roues s'enfoncer légèrement dans un sol mou tel que constitué de sable ou de boue par exemple (ou même sur une terre un peu meuble). Dans un MNT traditionnel la surface d'un terrain recouvert d'herbes sera positionnée pratiquement au niveau du sommet des herbes, alors que les roues du robot iront se poser au niveau de la terre, donc plutôt à la racine de ces herbes. À moins d'avoir une pelouse parfaitement tondue comme un terrain de football, ceci pourra donc générer des écarts entre un robot géométrique virtuel posé sur ce terrain et la réalité.

5.4.2 Principe

Les données mesurées (ou estimées à partir de mesures capteurs) à bord du robot et qui peuvent être prédites par la fonction de placement `PlaceRobot` sont :

– les angles d'attitude (dans le cas d'un robot ne disposant que d'une centrale inertielle il s'agit en fait d'une estimation, cf. annexe A.2),
– les angles de configuration internes pour les robots ayant des mobilités internes (Lama par exemple),
– l'altitude z du centre du robot.

Le fonctionnement de ce moniteur est le suivant. En ligne, une mesure des angles d'attitude et de configuration est effectuée périodiquement. Pour chacune de ces mesures, nous disposons d'une position (x, y, θ) correspondante et donc d'une position sur la grille du MNT. En cette position du MNT, nous faisons appel à la fonction `PlaceRobot` afin de déduire les angles d'attitude ϕ et ψ qui sont prédits. Si la comparaison entre la *prédiction* sur le MNT et l'*observation* est considérée comme acceptable (i.e. en tenant compte des erreurs connues sur cette prédiction), on pourra continuer à utiliser le mode *RoughNav* car son comportement est satisfaisant. Si au contraire cette différence n'est pas acceptable, cela signifie qu'il y a des erreurs importantes, probablement dues à l'une des sources d'erreurs mentionnées plus haut. On ne peut alors plus considérer le placement du robot sur le MNT actuel comme une méthode sûre de navigation.

En pratique, nous utiliserons uniquement les comparaisons des angles de roulis et de tangage, car ce moniteur est implémenté sur le robot Dala qui a un châssis rigide (en effet, ce moniteur est destiné à alimenter le système de sélection de modes de navigation embarqué précisément sur ce robot). D'autre part, la donnée d'altitude z est souvent entachée d'erreurs accumulées au cours du déplacement par les méthodes de localisation, ainsi la donnée dite "d'observation" est-elle trop incertaine pour servir de référence fiable.

Notons toutefois que l'analyse de cette comparaison entre observation en ligne et "prédiction" peut être réalisée en prenant également en considération les angles de configuration interne sur les robots comportant des mobilités internes (tels que Lama).

La figure 5.9 illustre un exemple de comparaison directe entre l'angle de tangage prédit par placement sur le MNT et celui mesuré, alors que le robot doit monter sur un bord de trottoir. La seule méthode de localisation utilisée dans cet exemple est l'odométrie 3D. Les différences constatées peuvent s'expliquer principalement ici par :

- l'erreur de localisation due à l'odométrie : les quelques dérapages intervenant principalement lors de la montée sur la marche des roues provoquent un décalage (suivant l'axe x de déplacement) entre prédiction et observation,
- la présence derrière le trottoir d'une surface un peu meuble (et recouverte d'herbes) qui est modélisée dans le MNT par une surface complètement rigide sur laquelle le robot est supposé pouvoir se poser sans provoquer de déformation.

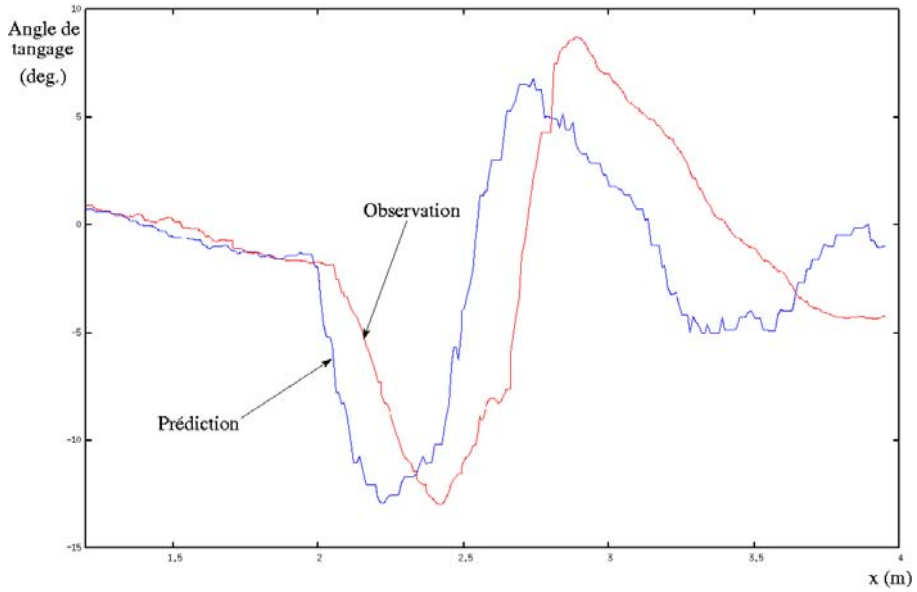


FIG. 5.9 – Différence entre angle de tangage prédit par placement sur le MNT et la mesure réalisée à bord du robot Dala. Les importantes variations d'attitude sont dues à la montée d'un bord de trottoir.

5.4.3 Mise en forme

Nous avons vu que des variances pouvaient être associées aux angles d'attitude du robot déduits de la prédiction de placement, soit le tangage ϕ et le roulis ψ. Par contre l'estimation d'attitude effectuée à partir de l'IMU de Dala ne fournit pas de modèle d'erreur. Étant néanmoins plus fiable, elle sera prise comme référence.

Cahier des charges

En tant qu'*expert* concepteur, nous précisons les caractéristiques d'allure de l'*indicateur* de comportement du mode *RoughNav* étudié ici, en ce qui concerne uniquement la surveillance ef-

fectuée par ce moniteur :

- Cet indicateur devra prendre la forme d'une "pseudo-probabilité", évoluant donc en particulier entre 0 (pour un comportement "parfait", i.e. exactement comme attendu) et 1 (pour un comportement des plus mauvais).
- Le comportement sera considéré comme bon (i.e. indicateur proche de 0) si la différence entre prédiction et observation reste inférieure aux erreurs prévues et évaluées lors de l'application de la fonction de placement du robot (cf. les σ_ϕ et σ_ψ).
- Cette différence sera d'autant moins tolérée que le robot sera proche des limites angulaires admissibles (stratégie conservatrice). En effet, un même écart de 5 degrés est potentiellement bien plus critique lorsque le robot est dans une zone de terrain particulièrement accidentée l'amenant à se rapprocher de ses limites ($|\phi_{obs}|$ proche de ϕ_{max}, ou $|\psi_{obs}|$ proche de ψ_{max}) que lorsqu'il est sur un terrain observé pratiquement plat.
- Il nous faut prendre en compte que des erreurs non modélisées par les σ du placement (voir plus haut) ne manqueront pas d'intervenir sur terrain accidenté naturel, dont certaines peuvent être tolérées. En effet, il s'agit de ne pas "crier au loup" trop rapidement et trop souvent. C'est pourquoi nous poserons un maximum de différence tolérable pour le comportement et définirons une fonction de mise en forme en conséquence.

Fonction de mise en forme

Considérant ce cahier des charges, décrivons la fonction de mise en forme appliquée à la différence prédiction-observation que nous avons posée et exploitée. Tout d'abord, afin de maîtriser le rapport aux limites angulaires, potentiellement différentes pour l'angle de tangage et celui de roulis, nous effectuons en premier lieu une mise en forme séparée pour chacun des angles ϕ et ψ.

Nous présentons ainsi cette fonction de mise en forme à partir de l'analyse de l'angle de tangage ϕ (le raisonnement est symétrique pour l'angle de roulis ψ). Indépendamment de l'influence de la proximité de l'angle observé avec l'angle limite, la fonction de comportement prend la forme d'une gaussienne normalisée dont l'argument est la distance $d_\phi = |\phi_{pred} - \phi_{obs}|$, de "moyenne" d_{max} (argument donnant la valeur maximale : 1) et d'écart-type σ_d. Soit :

$$g_\phi(d_\phi) = e^{-\frac{1}{2}\left(\frac{d_\phi - d_{max}}{\sigma_d}\right)^2}$$

Sachant que la Gaussienne de moyenne m et d'écart-type σ contient 99% de sa densité dans un intervalle de largeur $3\,\sigma$ de chaque côté de m, et que l'on considère que le comportement est bon (i.e. pseudo-probabilité de comportement proche de 0) lorsque d_ϕ est nul, nous posons : $3\,\sigma_d = d_{max} - 3\,\sigma_\phi$, d'où $\sigma_d = (d_{max} - 3\,\sigma_\phi)/3$ (figure 5.10).

Influence de la limite angulaire Nous ajoutons à cela l'influence du rapport $|\phi_{obs}|/\phi_{max}$, i.e. la proximité de limite angulaire admissible. Nous choisissons également une courbe d'influence de type gaussienne, de "moyenne" ϕ_{max} et avec le même σ, pour appliquer une influence de type exponentielle (avec une influence quasi-nulle lorsque l'angle observé est proche de 0).

La fonction de comportement complète devient alors :

$$f_\phi(d_\phi, \phi_{obs}) = e^{-\frac{1}{2}\left(\frac{d_\phi - d_{max}}{\sigma_d}\right)^2} + e^{-\frac{1}{2}\left(\frac{\phi_{obs} - \phi_{max}}{\sigma_d}\right)^2}\left(1 - e^{-\frac{1}{2}\left(\frac{d_\phi - d_{max}}{\sigma_d}\right)^2}\right)$$

La figure 5.11 montre un tracé de cette fonction de deux variables donnant une grandeur de pseudo-probabilité de mauvais comportement du ϕ pour le mode *RoughNav*.

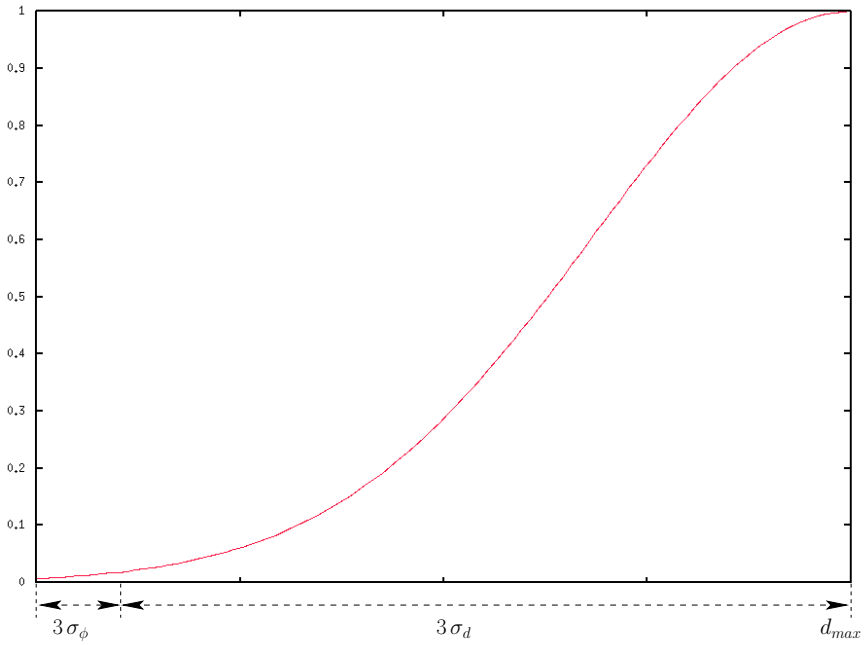


FIG. 5.10 – Pseudo-probabilité de comportement en fonction de d_ϕ.

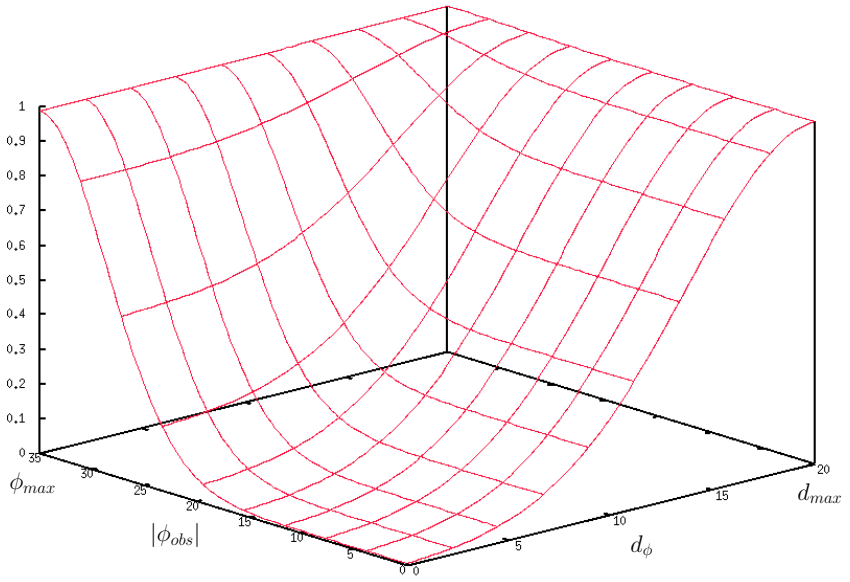


FIG. 5.11 – Pseudo-probabilité de comportement avec l'influence de la limite angulaire ϕ_{max}.

La fonction de mise en forme pour ψ s'obtient de la même manière :

$$f_\psi(d_\psi, \psi_{obs}) = e^{-\frac{1}{2}\left(\frac{d_\psi - d_{max}}{\sigma_d}\right)^2} + e^{-\frac{1}{2}\left(\frac{\psi_{obs} - \psi_{max}}{\sigma_d}\right)^2}\left(1 - e^{-\frac{1}{2}\left(\frac{d_\psi - d_{max}}{\sigma_d}\right)^2}\right)$$

Distance de Mahalanobis

Afin de comparer la *prédiction* et l'*observation* en prenant en compte les erreurs "prévues", nous pourrions utiliser la distance de Mahalanobis. Dans un cas général, la distance de Mahalanobis entre un vecteur connu $\boldsymbol{y}$ et des données incertaines de vecteur moyen $\boldsymbol{x}$ et de matrice de covariances Σ s'exprime par :

$$D_M(y) = \sqrt{(\boldsymbol{y} - \boldsymbol{x})^T \Sigma^{-1} (\boldsymbol{y} - \boldsymbol{x})}$$

Si la matrice de covariances est diagonale, ne comportant donc que les variances, elle peut s'écrire :

$$d_M(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i \frac{(x_i - y_i)^2}{\sigma_i^2}}$$

où les x_i et y_i sont les composantes des vecteurs $\boldsymbol{x}$ et $\boldsymbol{y}$ et où les σ_i sont les écarts-types sur les composantes du vecteur incertain $\boldsymbol{x}$.

Exprimons cette distance de Mahalanobis entre le vecteur dit d'*observation* $\boldsymbol{y} = [\phi_{obs}\ \psi_{obs}]^T$ et celui dit de *prédiction* $\boldsymbol{x} = [\phi_{pred}\ \psi_{pred}]^T$, avec les écarts-types associés σ_ϕ et σ_ψ.

$$d_M(\mathbf{x}, \mathbf{y}) = \sqrt{\frac{(\phi_{pred} - \phi_{obs})^2}{\sigma_\phi^2} + \frac{(\psi_{pred} - \psi_{obs})^2}{\sigma_\psi^2}}$$

La densité de probabilité que ces deux données correspondent est alors proportionnelle à : $e^{-d_M^2/2}$.

5.4.4 Illustration

La figure 5.12 présente le résultat de l'expérimentation vue en figure 5.9 après application de la fonction de mise en forme. On peut y constater en particulier un moment où la pseudo-probabilité de comportement prend une valeur significative (i.e. se rapprochant de 1), révélant un problème de placement.

5.5 Moniteurs additionnels

Des moniteurs additionnels peuvent être envisagés pour compléter la surveillance de comportement des modes de déplacement utilisés. Il est intéressant d'en avoir autant que possible pour chaque mode de manière à disposer d'informations supplémentaires (qu'il conviendra de combiner) et d'augmenter la redondance, ce qui peut permettre d'obtenir une surveillance plus efficace.
Certains moniteurs que nous proposons ici sont en cours d'étude, en vue d'une proche intégration, d'autres au contraire ne sont encore qu'au stade de projets.

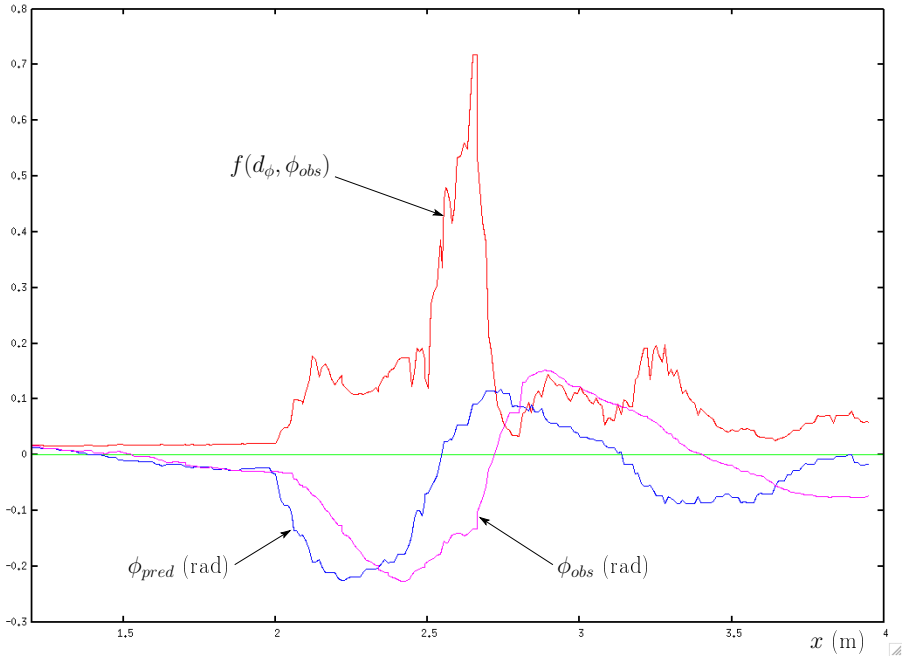


FIG. 5.12 – Pseudo-probabilité de comportement obtenue après mise en forme de la différence d_ϕ dans le même exemple que la figure 5.9 : le robot Dala monte sur un bord de trottoir. L'écart le plus important entre prédiction et observation est constaté lorsque les roues arrières du robot montent à leur tour le bord de trottoir et que le robot "pique du nez" plus que prévu car ses roues avant s'enfoncent un peu dans le sol constitué d'herbes sur terre meuble.

5.5.1 Moniteur de localisation (*Locomp*)

Ce moniteur effectue des comparaisons entre des méthodes de localisation différentes. Le fonctionnement est nominal concernant la localisation lorsque les estimées de la position fournies par ces différentes méthodes sont comparables. Des écarts entre ces différentes estimées peuvent permettre de révéler des problèmes de comportement du robot. Par exemple, si l'on constate des écarts entre les estimations de déplacement relatif de l'*odométrie 3D* (annexe A.1) et de l'*odométrie optique* (estimation visuelle de mouvement, annexe A.3), cela signifie généralement que le robot subit des glissements au sol, traduisant des problèmes de locomotion.

5.5.2 Autres moniteurs

Voici quelques exemples d'autres possibilités de moniteurs que nous avons envisagés :

- la *surveillance des accélérations mesurées par l'IMU*. En utilisant par exemple un modèle dynamique du robot, cette surveillance pourrait permettre de vérifier que les déplacements se réalisent comme prévu ;
- l'*évaluation de la qualité de la calibration du banc de stéréo-vision* (i.e. taux de réussite des appariements de pixels, précision de la corrélation) [Lemonde, 2005]. En effet, si le banc n'est plus correctement calibré, aucun mode de navigation ne pourra se fier aux données de stéréo-vision pour construire un MNT par exemple ;
- l'*évaluation de l'énergie consommée* par rapport à des conditions nominales (voir [Grand, 2004]) ;
- la *surveillance de l'évolution de la position du robot relativement aux obstacles*, afin de vérifier que le déplacement se déroule comme prévu, avec la trajectoire souhaitée, d'après les données de perception ;
- la *cohérence entre les obstacles détectés par différents moyens de perception* (e.g. la stéréo-vision et le laser). Si par exemple la stéréo-vision repère un obstacle qui ne figure pas dans la carte locale des obstacles issus des données de réflexion du laser, cela signifie qu'il y a danger à utiliser un mode ne se reposant que sur le laser pour décider quel mouvement effectuer ;
- la *marge de stabilité*, en particulier pour les structures complexes (voir annexes, section B.4.1) ;
- pour le mode de suivi de chemin, la *qualité d'extraction du chemin* dans l'image par le processus de classification. En effet, on peut exploiter les résultats quantitatifs de la classification pour savoir si la zone considérée comme un chemin se distingue nettement des autres zones de l'image et aussi à quel point les données de cette zone sont semblables aux prototypes de la base d'apprentissage correspondant à la classe *Chemin*.

5.6 Synthèse des moniteurs

Le tableau suivant présente une synthèse des moniteurs présentés, précisant pour chacun :

- les modes de navigation/locomotion dont ils surveillent le comportement,
- le robot sur lequel ce moniteur peut être utilisé,
- les données exploitées (capteurs).

Moniteur	Mode	Robot	Données
LEM	*Roulement*	Dala & Lama	V, ω, variations d'attitude et de configuration
DAT	*FlatNav*	Dala (voire Lama)	ω_x, ω_y ($\ddot{z}$)
SurvAtt	*RoughNav*	Dala & Lama	(ϕ_{obs},ψ_{obs}) vs. $(\phi_{pred},\psi_{pred})$
Locomp	Tous	Tous	Estimées de localisation

5.7 Conclusion

Nous avons vu dans ce chapitre différents *moniteurs de comportement* que nous avons développés. Ces moniteurs ont pour rôle de surveiller le comportement d'un mode de navigation actif en comparant des signatures produites à des signatures de référence prédéfinies (comportement nominal). Il fournissent une valeur quantitative de comportement, normalisée, correspondant à une probabilité ou à défaut une *pseudo-probabilité* au comportement comparable. Ce sont ces grandeurs qui alimentent le *modèle dynamique* et donc les probabilités de transitions du sytème de sélection de mode introduit au chapitre 2. Nous verrons dans le prochain chapitre comment les moniteurs qui viennent d'être décrits s'intègrent dans les systèmes multi-modes proposés section 3.4.

Bien entendu, d'autres moniteurs peuvent être envisagés pour les différents modes (nous en avons proposé quelques possibilités, mais bien d'autres encore pourraient être considérés) et leur ajout permettrait d'augmenter le niveau d'informations sur lesquelles on raisonne, et donc assurément d'améliorer le comportement général du robot pour la réalisation de sa tâche de déplacement.

Certains moniteurs sont spécifiques à un mode, surveillant des caractéristiques qui lui sont propres (par exemple le moniteur de Surveillance d'Attitude, *SurvAtt*, qui évalue la qualité de la fonction de placement sur laquelle repose en bonne partie le mode de navigation *RoughNav*). D'autres moniteurs sont au contraire plus génériques, lorsqu'ils exploitent des signatures qui peuvent être générées dans plusieurs modes différents (c'est le cas par exemple du moniteur de localisation *Locomp* car les méthodes de localisation surveillées sont applicables quelque soit le mode de déplacement utilisé).

Ainsi, nous voyons apparaître en particulier le besoin de *combiner* les informations de plusieurs moniteurs surveillant un même mode, afin de prendre en compte les données de comportement "complètes" dans les transitions. De même que les moniteurs utilisés dépendent des modes de déplacement disponibles et que ces derniers dépendent de la plate-forme robotique, la manière d'effectuer cette combinaison de données et de prendre en compte le comportement dans les transitions va dépendre du système multi-modes mis en place. Ce point sera donc abordé plus spécifiquement dans le chapitre suivant, traitant de l'intégration complète des systèmes de sélection de modes de navigation et de locomotion proposés dans le chapitre 3.

Chapitre 6

Intégration et expérimentations

Nous présentons dans ce chapitre l'intégration et des expérimentations réalisés avec les deux systèmes multi-modes proposés à la fin du chapitre 3 (section 3.4), soit un système multi-modes de locomotion conçu pour être embarqué sur un robot tel que Lama et un système multi-modes de navigation pour un robot tel que Dala. Les expérimentations et l'étude des résultats sont surtout consacrées à ce dernier.
Nous présentons progressivement les différentes composantes de l'ensemble des outils d'estimation de mode à appliquer, en les intégrant les unes après les autres afin de mieux identifier leur rôle et leur influence. Enfin, nous proposons une discussion sur ces résultats et les enseignements qui en sont tirés.
L'intégration logicielle de ces éléments sur les robots fait quant à elle l'objet de l'annexe C.

6.1 Système multi-modes de locomotion de Lama

Le système multi-modes de locomotion pour Lama (figure 6.1) repose sur deux modes "actifs" : le *Roulement* et le *Péristaltisme*, plus le "non-mode" *Stop* (voir section 3.4.1 page 68). Nous ne disposons ici que d'un moniteur pour surveiller le comportement du mode de roulement : le *LEM* (Locomotion Efficiency Monitoring, voir section 5.2 page 86), qui évalue l'efficacité de la locomotion. Nous pourrions néanmoins imaginer la mise au point d'un moniteur évaluant d'une manière similaire la locomotion lors du péristaltisme (que nous pourrions nommer *PLEM*). C'est dans cette hypothèse que se place la figure 6.1.

6.1.1 Observation du contexte

Les contextes correspondant aux modes sont :
- Terrain cohésif franchissable pour le mode *Roulement*,
- Terrain peu cohésif mais franchissable pour le *Péristaltisme*,
- Obstacle (i.e. terrain non franchissable) pour le mode *Stop*.

Savoir distinguer ces contextes nécessiterait des moyens d'observation effectuant une étude de la texture du sol (la vision couleur semble a priori la méthode la plus adaptée). Or nous ne disposons pas encore à l'heure actuelle de tels moyens opérationnels sur nos robots, malgré les premiers travaux sur la classification de texture embarquée réalisés dans cet objectif par nos collègues du LASMEA dans le cadre du projet commun R2M (voir annexe B.1), ou encore les quelques études sur la texture de [Aviña, 2005] effectuées au sein de notre laboratoire.

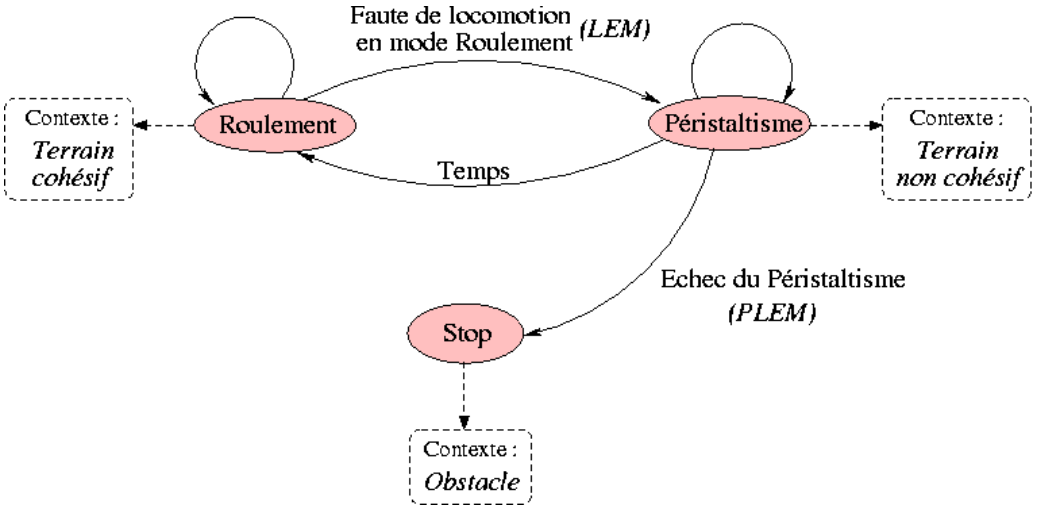


FIG. 6.1 – Système multi-modes de locomotion pour Lama, exploitant les modes *Roulement* avec TALC (pour des sols cohésifs) et *Péristaltisme* (surtout pour les sols peu cohésifs). Les états (noeuds) sont les *modes à appliquer*.

6.1.2 Comportement et moniteurs

Le moniteur *LEM* traduit l'efficacité de la locomotion du mode de *Roulement*. Lorsque ce mode n'est pas efficace, l'autre option à utiliser est l'exploitation du mode de *Péristaltisme*, puisqu'il permet précisément au robot de bénéficier d'une meilleure adhérence et donc d'une locomotion plus efficace sur des terrains peu cohésifs. Ainsi, la transition de l'état "le mode *Roulement* est le mode à appliquer" au temps $(t-1)$ à l'état adjacent "le mode *Péristaltisme* est le mode à appliquer" au temps t sera influencée par l'indicateur de comportement fourni par le *LEM*. Notons R le premier mode et P le deuxième. L'équation 2.10 page 41 s'écrit alors, pour cette transition :

$$P(x_{P,t}|x_{R,t-1}, C_{1:t}) = P(x_{P,t}|x_{R,t-1}) + (1 - P(x_{P,t}|x_{R,t-1}))Q_{R,P}(C_{1:t})$$

La probabilité de transition "a priori" $P(x_{P,t}|x_{R,t-1})$ (c'est à dire sans prise en compte d'informations de comportement) prend une valeur empirique fixée *a priori* qui dépendra essentiellement de deux éléments :

- le "design" du modèle d'observation du contexte, permettant d'évaluer à partir des données de stéréo-vision si le terrain semble être de type cohésif ou non,
- la "sensibilité" souhaitée pour la sélection de modes, à savoir le niveau d'évidence qu'il faudra accumuler sur un mauvais comportement de la locomotion en mode *Roulement* avant que le mode *Péristaltisme* soit effectivement "favorisé".

La loi d'influence du comportement que nous choisissons d'adopter est la suivante : le mode à appliquer à t est P plutôt que R à $(t-1)$ lorsqu'il y a preuve d'une *faute de locomotion*. Autrement dit, si nous notons *FauteLoco* l'état s_2 défini dans la section 5.2 (page 86) lors de la description du *LEM*, la transition $(x_{P_t}|x_{R_{t-1}})$ est caractérisée par la probabilité qu'il y ait une faute de locomotion :

$$Q_{R,P}(C_{1:t}) = P(FauteLoco|obs_{1:t})$$

De plus, par définition du HMM, pour un état donné, la somme des probabilités des transitions sortantes doit être égale à 1. Par conséquent, la probabilité de transition conditionnelle de l'état

x_R à x_R (i.e. maintien du mode de roulement) s'exprime :

$$P(x_{R,t}|x_{R,t-1}, C_{1:t}) = 1 - P(x_{P,t}|x_{R,t-1}, C_{1:t})$$

Étudions maintenant la transition du mode *Péristaltisme* vers le *Roulement*. De manière qualitative, il s'agit du "retour" au mode de roulement qui est le mode par défaut (rappelons que le péristaltisme est bien plus lent que ce dernier et qu'il y est difficile de tourner, voire impossible avec certains types de châssis). Le modèle de comportement envisagé est donc tel que, en dehors des considérations de contexte, l'on puisse revenir vers le mode de roulement après un laps de temps spécifié Δt.

6.1.3 Illustration

La figure 6.2 illustre un exemple de probabilités issues de l'estimation du mode de locomotion à appliquer sur Lama (concernant en particulier les modes "actifs" *Roulement* et *Péristaltisme*) lors de l'apparition d'une faute de locomotion en roulement. Les données de locomotion exploitées pour effectuer le monitoring ont été récoltées en ligne lors d'un test réel sur le robot. Les probabilités de mode ont par contre été générées hors-ligne, le robot Lama n'étant désormais plus opérationnel pour réaliser de nouvelles expérimentations.

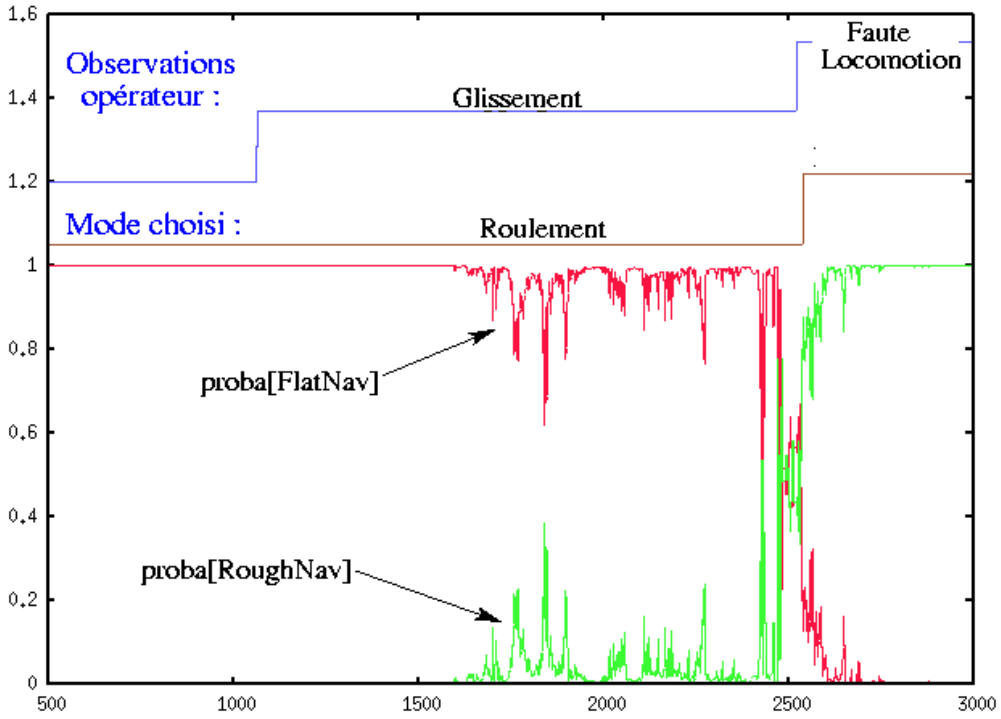


FIG. 6.2 – Exemple de résultat de l'estimation du mode de locomotion à appliquer pour le robot Lama, en particulier entre *Roulement* et *Péristaltisme*, et la sélection de mode qui en découlerait. Pour illustrer le comportement réel du robot, un opérateur humain donne en même temps ses observations.

6.2 Système multi-modes de navigation de Dala

6.2.1 Fonctionnement du système multi-modes

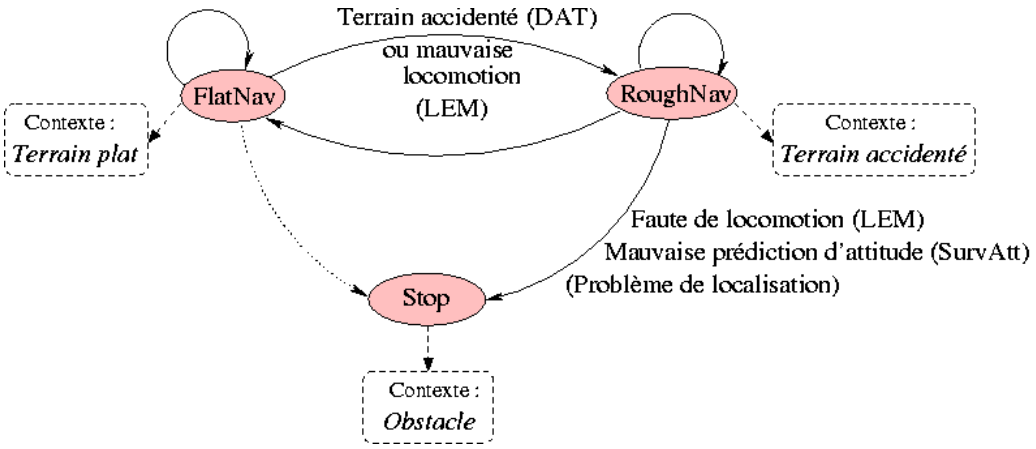


FIG. 6.3 – Système multi-modes de navigation pour Dala, exploitant les modes dits *FlatNav* (valide seulement sur terrain plat) et *RoughNav* (dédié aux terrains accidentés). Les états (noeuds) sont les *modes à appliquer*.

Nous proposons tout d'abord d'étudier "intuitivement" le rôle des moniteurs que nous utilisons dans cette sélection de mode de navigation pour Dala. Comme nous l'avons vu à la fin du chapitre 3, nous exploitons les modes de navigation suivants :

- *FlatNav* (section 3.2.1) : utilisable *uniquement sur terrain plat*, il exploite une méthode de mouvement réactif sur la base de mesures laser 2D,
- *RoughNav* (section 3.2.2) : plutôt dédié aux terrains accidentés (mais suffisamment générique pour pouvoir s'appliquer aussi en terrain plat), il exploite la fonction de placement du robot sur le MNT construit par stéréo-vision,
- le "non-mode" *Stop* : toute navigation y est arrêtée, car le robot est considéré comme ne disposant pas de moyen de traverser la zone de terrain qui se présente à lui, du moins dans la situation actuelle.

Le mode *Stop* n'a bien entendu pas de moniteurs associés, vu qu'il n'y a plus d'action dans ce mode et qu'il n'y a aucune transition sortante.

Le mode *FlatNav* est surveillé par le moniteur *DAT* (section 5.3 page 91) et par l'évaluateur d'efficacité de la locomotion, le *LEM* (section 5.2 page 86). Si une difficulté de locomotion ou un accident de terrain son détectés respectivement par le *LEM* ou le *DAT*, cela signifie que le terrain n'est pas aussi plat et facile que le système pouvait le penser en étudiant seulement les données de la carte qualitative du terrain. Dès lors, il convient de favoriser le mode *RoughNav* qui prend en compte une modélisation de l'environnement plus complète pour planifier les déplacements : la transition de l'état $x_{FlatNav}$ (i.e. "le mode à appliquer est *FlatNav*") vers l'état $x_{RoughNav}$ (i.e. "le mode à appliquer est *RoughNav*") doit être favorisée.

Le mode *RoughNav* est également surveillé par le *LEM*, mais aussi par le moniteur *SurvAtt* (section 5.4) auxquels nous pourrions ajouter le moniteur de localisation *Locomp* (qui n'est actuellement pas encore opérationnel mais envisagé à court terme dans notre étude, voir 5.5.1). Comme nous l'avons vu, le moniteur *SurvAtt* étudie la qualité du placement du robot sur le MNT par rapport aux observations, il étudie donc bien directement le comportement du mode

RoughNav. Si ce comportement est mauvais, il convient de trouver une autre solution si l'on en a une disponible, autrement dit il convient de favoriser la transition vers un autre mode. En pratique, si le mode *RoughNav* est celui qui est actif, cela signifie que :

- ou bien le terrain est accidenté (d'après les informations d'observation du terrain qui figurent sur la carte qualitative),
- ou bien un problème de comportement (mauvaise locomotion par exemple) a amené le système à choisir ce mode de terrain accidenté plutôt que *FlatNav*.

Par conséquent, un problème de comportement de *RoughNav* ne doit pas rediriger le système vers un mode tel que *FlatNav*. Dès lors, n'ayant pas d'autre solution dans ce système, cela ne peut mener que vers le mode *Stop*, pour tout arrêter. Le superviseur de haut niveau sera alors chargé de déterminer que faire dans ce cas de figure (déclarer un problème et ne pas aller plus loin, ou trouver un moyen de poursuivre le mouvement, par exemple en réinitialisant le modèle de l'environnement et la localisation).

Le moniteur de localisation peut ici être actif avec n'importe quel mode, vu que les méthodes de localisation surveillées sont utilisées indifféremment du mode de navigation actif. Cela étant, une mauvaise localisation aura moins de conséquences directes sur le comportement d'un mode tel que *FlatNav*, dans lequel le mouvement est réalisé de manière purement *réactive* : en fonction du positionnement relatif des obstacles par rapport au robot. Dans ce cas, la qualité de la localisation aura donc simplement une influence sur l'étude du positionnement de l'objectif (local, voire global) par rapport au robot et donc aussi sur le choix de la *situation* au sens défini dans la description du mode *FlatNav*, section 3.2.1. Au contraire, un problème de localisation a des conséquences fondamentales sur le mode *RoughNav*. En effet, si la localisation est incorrecte, le robot ne pourra plus réaliser un modèle de l'environnement fidèle à la réalité du terrain, on ne peut donc plus garantir une planification correcte du mouvement, évitant en particulier les zones non franchissables : le résultat du placement du robot sera trop entaché d'erreur. Nous voyons apparaître ici un lien causal entre un problème de comportement de localisation et de comportement du placement (normalement révélé par le moniteur *SurvAtt*). Il faudra bien entendu prendre en compte cette relation causale dans la combinaison des moniteurs intervenant dans la transition de *RoughNav* vers *Stop*, si l'on veut intégrer le moniteur de localisation.

La transition de *RoughNav* vers *Stop* sera donc favorisée en cas de :

- mauvais comportement révélé par le moniteur *Attitude Watching*,
- faute de locomotion, révélée par le *LEM*,
- mauvaise localisation (d'après le moniteur *Locomp*, en perspective à court terme).

Probabilités de transition

Probabilités de transition *a priori* Les probabilités de transition *a priori* qui ont été posées en pratique pour le système multi-modes de navigation de Dala sont illustrées dans la figure 6.4.

Probabilités de transition "complètes" Rappelons que l'équation 2.10 page 41 nous donnait l'expression de la probabilité de transition de l'état x_c vers l'état x_k (avec $k \neq c$) lorsque l'on dispose de données de comportement pour le mode m_c sous la forme des $Q_{c,k}(C_{1:t})$:

$$P(x_{k,t}|x_{c,t-1}, C_{1:t}) = P(x_{k,t}|x_{c,t-1}) + (1 - P(x_{k,t}|x_{c,t-1}))Q_{c,k}(C_{1:t})$$

Notation : dans un souci de simplification, les états $x_{FlatNav}$, $x_{RoughNav}$ et x_{Stop} dans les expressions probabilistes seront simplement notés dans la suite de ce chapitre respectivement *FlatNav*, *RoughNav* et *Stop*, voire *FN*, *RN* et *S*. Ainsi nous avons $P(x_{FlatNav}) = P(FlatNav) =$

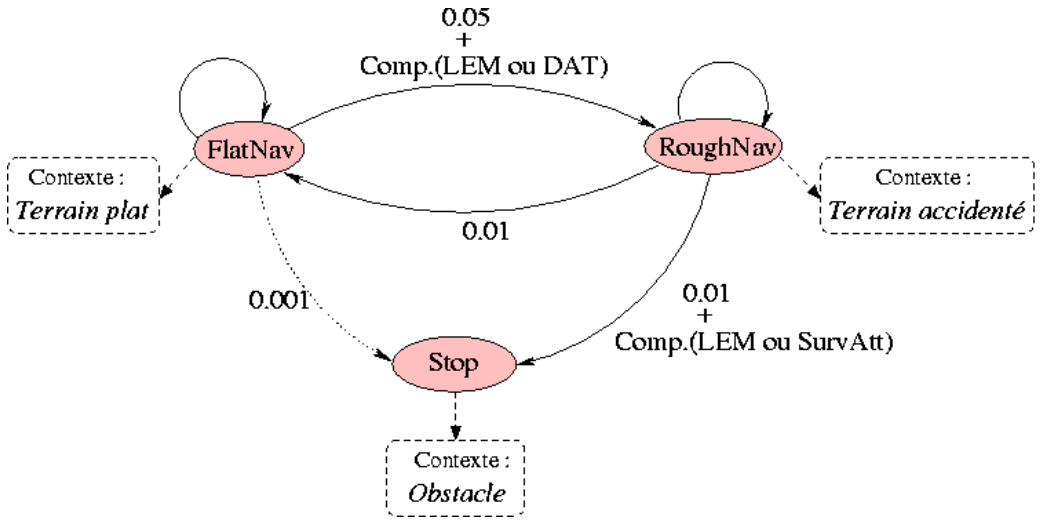


FIG. 6.4 – Probabilités de transitions pour le système multi-modes de navigation de Dala, constituées des probabilités "a priori" fixées (cf. les étiquettes chiffrées sur les arcs) plus l'éventuelle contribution des données de comportement.

$P(FN)$.
Considérons ici le cas de la probabilité (sachant le comportement) de la transition de l'état $x_{FlatNav}$ (le mode à appliquer est *FlatNav*) vers l'état $x_{RoughNav}$ (le mode à appliquer est *RoughNav*). Elle s'écrit, avec la notation simplifiée :

$$P(RN|FN,C) = P(RN|FN) + (1 - P(RN|FN))Q_{FN,RN}(C)$$

où $P(RN|FN)$ est la probabilité "a priori" de transition de FN à RN et $Q_{FN,RN}(C)$ la contribution des données de comportement (il s'agit d'une pseudo-probabilité de comportement). Ainsi, dans le cas où l'information de contexte alimentant une transition ne provient que d'un seul moniteur (*LEM* par exemple dans le cas de la transition que nous étudions ici), $Q_{FN,RN}(C)$ est la probabilité (ou pseudo-probabilité) de mauvais comportement d'après ce moniteur. Par exemple, la probabilité d'avoir un cas de glissement (cf. section 5.2) est :

$$Q_{FlatNav,RoughNav}(C) = Q_{FN,RN}(C) = P(Glissement|obs) = P(s_1|obs)$$

Moniteurs *DAT* et *LEM* pour le comportement de *FlatNav* Notons que les données de comportement alimentant la probabilité de transition $P(RoughNav|FlatNav,C)$ (provenant du moniteur *DAT* ou de *LEM*) sont calculables quand le mode *FlatNav* est activé mais aussi quand c'est un autre mode, tel que *RoughNav*. Elles donnent alors des informations sur le comportement que *pourrait avoir FlatNav* ("comportement virtuel"). L'influence de ces deux moniteurs sur la transition de $x_{FlatNav}$ à $x_{RoughNav}$ sera donc maintenue en permanence, même si le mode actif est en pratique *RoughNav*.

6.2.2 Expérimentations avec moniteur seul

Les tests de cette section ont été réalisés en récoltant les données pour les moniteurs et les observations à bord du robot, en ligne, mais les probabilités de modes ont été calculées hors-ligne

(les changements de mode effectifs correspondants n'ont donc pas été réalisés en pratique à bord du robot). Ces tests font intervenir des problèmes de comportement repérés par un moniteur seulement. Les données de contexte (les $p(obs|etat)$) sont fixées par l'opérateur (en pratique dans cette section les probabilités correspondantes seront maintenues à la même valeur pendant tout le test). Nous proposons divers exemples afin de souligner en particulier le rôle des moniteurs lorsque les données de comportement "contredisent" les indications données par les informations de contexte.

La figure 6.5 reprend ainsi le cas expérimental présenté dans la figure 5.7 page 94, lors de la présentation du moniteur *DAT*.

La figure 6.6 présente les probabilités d'application des modes et la probabilité de transition de *RoughNav* vers *Stop* lors d'une montée de trottoir similaire au test de la figure 5.12 page 101 mais avec des écarts prédictions/observations plus importants, amenant une demande de changement de mode.

Dans le cas de la figure 6.6, correspondant exactement au test déjà introduit dans la figure 5.12 page 101, à savoir une autre montée de trottoir mais générant moins d'erreurs, la demande de changement de mode ne s'effectue pas car les données de comportement restent relativement faibles.

La figure 6.8 présente un cas similaire (montée de trottoir) pour lequel nous avons introduit une erreur de localisation systématique de 0,5 m en x_{Robot}, soit selon la direction du mouvement (ce décalage en position est visible sur les courbes d'angle de roulis prédit et observé, clairement décalées). Cela provoque un écart manifeste de l'attitude entre prédiction et observation dès que le robot aborde le trottoir "virtuel", incitant ainsi à un basculement en mode *Stop* dès que ce dernier est abordé.

6.2.3 Moniteurs combinés et observations fixées par un opérateur

De la combinaison de moniteurs

Comme nous avons déjà pu le voir, si certains moniteurs sont spécifiques à la surveillance d'un mode en particulier, d'autres sont plus génériques, pouvant ainsi s'appliquer à la surveillance de plusieurs modes. En utilisant l'une et/ou l'autre de ces catégories, on peut dans certains cas avoir des modes surveillés par plusieurs moniteurs, plus ou moins complémentaires. Afin de déterminer une probabilité de comportement exploitable pour mettre à jour le modèle de transition, il convient alors de *combiner* ces moniteurs.

Grâce à "la mise en forme" des données issues des moniteurs, nous disposons d'un *indicateur* de comportement pour chacun d'eux, normalisé (il peut prendre des valeurs entre 0 et 1) et sous la forme d'une probabilité voire d'une pseudo-probabilité. Les indicateurs ont ainsi, grâce à cette opération, des comportements comparables qui vont faciliter leur combinaison.

Deux stratégies principales de combinaison des moniteurs ont été considérées :

- La première est de combiner directement les indicateurs (qui peuvent même éventuellement ne pas avoir été normalisés et mis en forme au préalable) pour générer une probabilité de comportement d'un mode à partir d'un vecteur d'attributs de dimension égale au nombre d'indicateurs disponibles pour la surveillance de ce mode (i.e. le nombre de moniteurs). Ce type de combinaison donne une grande latitude sur la manière de combiner les informations, suivant les liens (causaux, logiques) qui peuvent exister entre ces données.
- La deuxième est de "fusionner" les probabilités déjà calculées pour chacun des moniteurs. Cela permet de conserver plus de modularité et de flexibilité dans la manipulation des

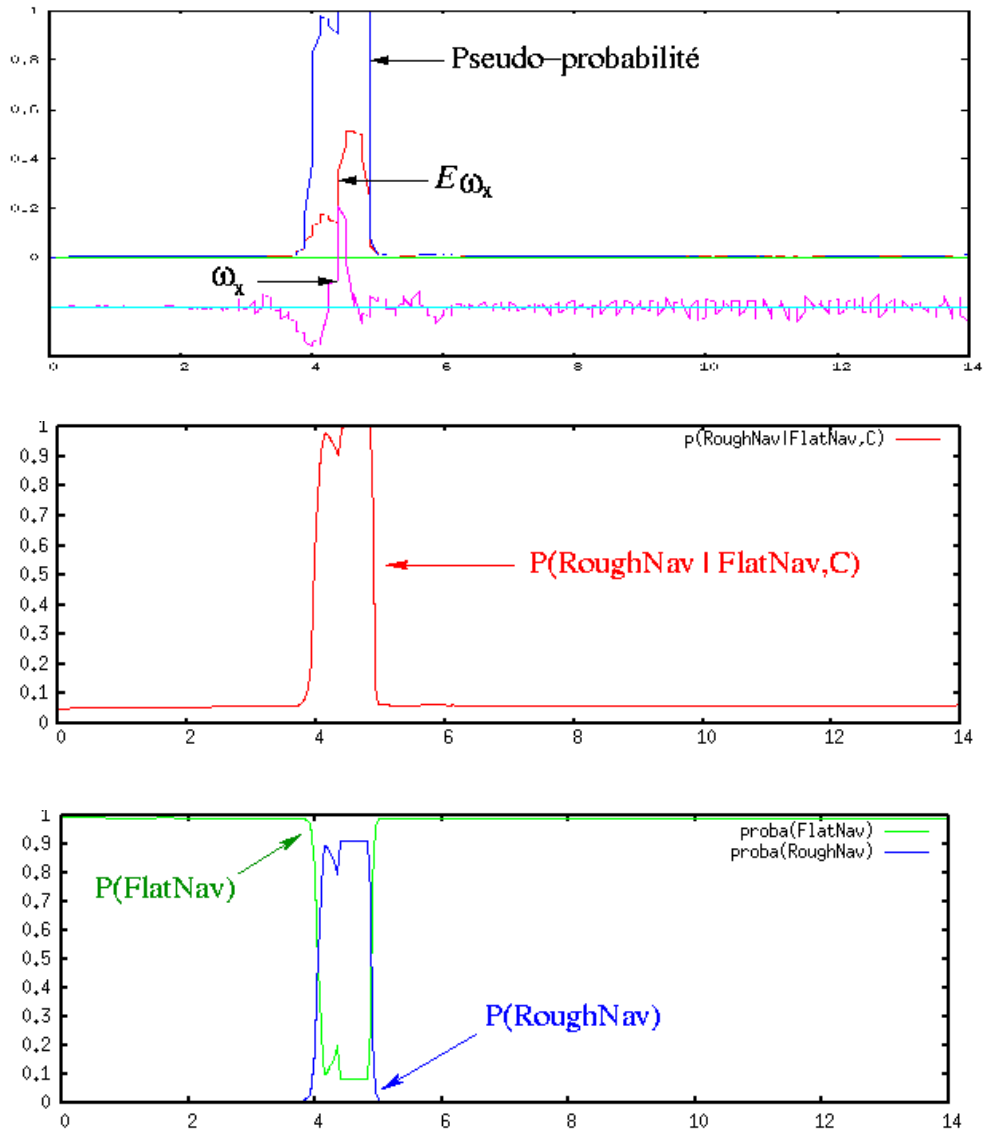


FIG. 6.5 – **Moniteur *DAT* :**
Probabilités d'application des modes et probabilité de transition de *FlatNav* vers *RoughNav* lors du test introduit dans la figure 5.7 page 94. Le robot circule sur un terrain plat en dehors d'une pierre qu'il rencontre vers $t = 4s$.
Les données de contexte sont fixées à : $p(O|Plat) = p(O|FlatNav) = 0.9$, $p(O|Accidente) = p(O|RoughNav) = 0.09$ et $p(O|Obstacle) = p(O|Stop) = 0.01$, pour signifier que l'opérateur voit un terrain bien plat et n'a pas vu cette pierre (ou a estimé qu'elle ne posait pas de problème pour le déplacement du robot en mode *FlatNav*). Les probabilités initiales sont quant à elles : $P(FlatNav) = 0.9$, $P(RoughNav) = 0.09$ et $P(Stop) = 0.01$.
On peut voir que la détection d'accident de terrain réalisée par le moniteur *DAT* provoque une "recommendation" de changement de mode vers *RoughNav*, le mode de terrain accidenté, en dépit de l'observation fournie par l'opérateur indiquant que le terrain semblait être bien plat (ce qui justifiait l'utilisation du mode *FlatNav*). Ce changement est toutefois temporaire car le système tend vite à souhaiter le retour au mode de terrain plat une fois la pierre franchie.

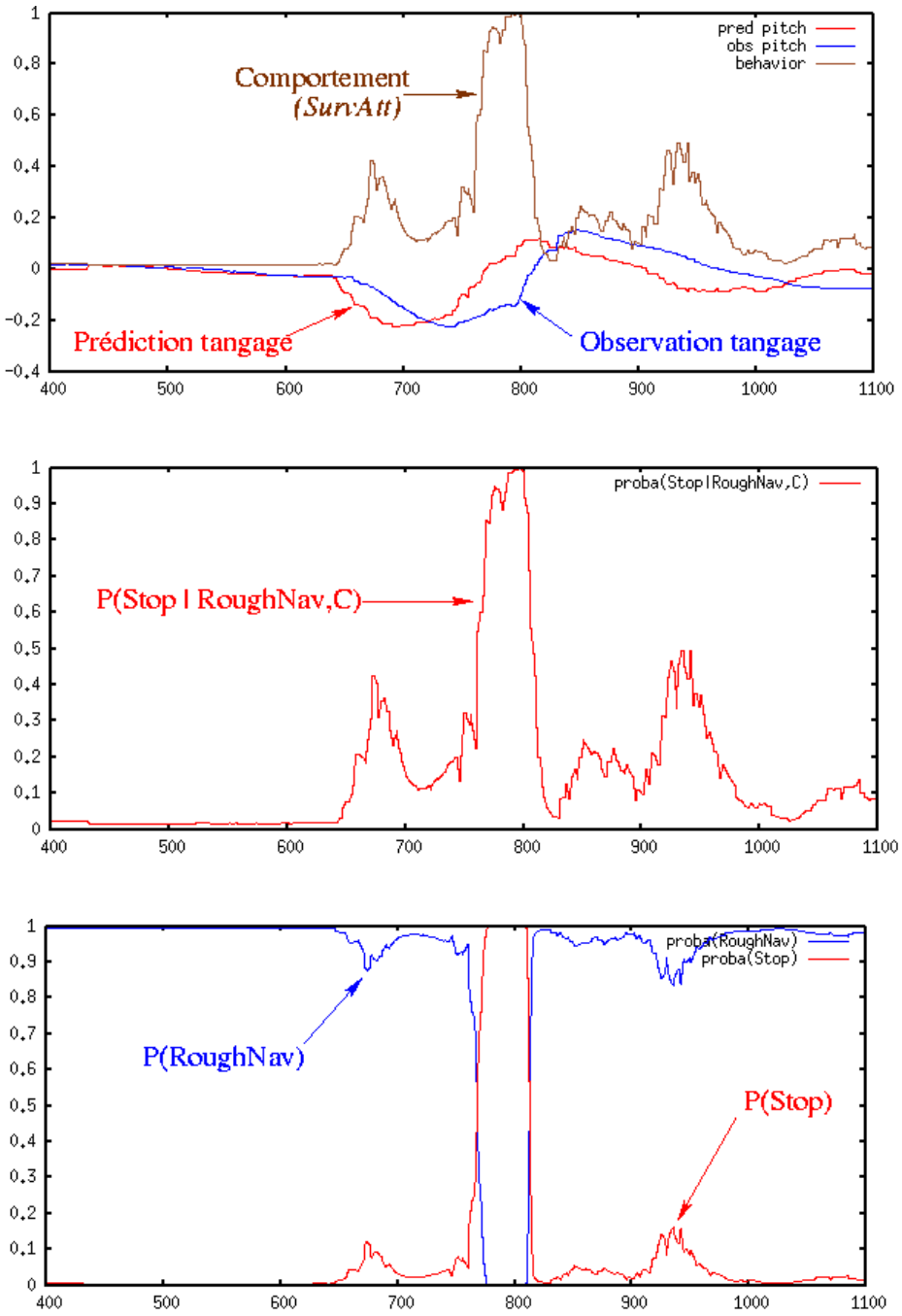


FIG. 6.6 – **Moniteur *SurvAtt* :**
Probabilités d'application des modes et probabilité de transition de *RoughNav* vers *Stop* lors d'une montée de trottoir. Un problème de comportement révélé par le moniteur *SurvAtt* (écart entre prédiction et observation de l'attitude) amène à une demande de changement de mode. Le système ne proposant pas d'autre solution en terrain accidenté, c'est le mode *Stop* qui est considéré comme celui à appliquer.
Les données de contexte sont ici fixées à : $p(O|Plat) = p(O|FlatNav) = 0.09$, $p(O|Accidente) = p(O|RoughNav) = 0.8$ et $p(O|Obstacle) = p(O|Stop) = 0.11$; et les probabilités initiales sont : $P(FlatNav) = 0.09$, $P(RoughNav) = 0.9$ et $P(Stop) = 0.01$.

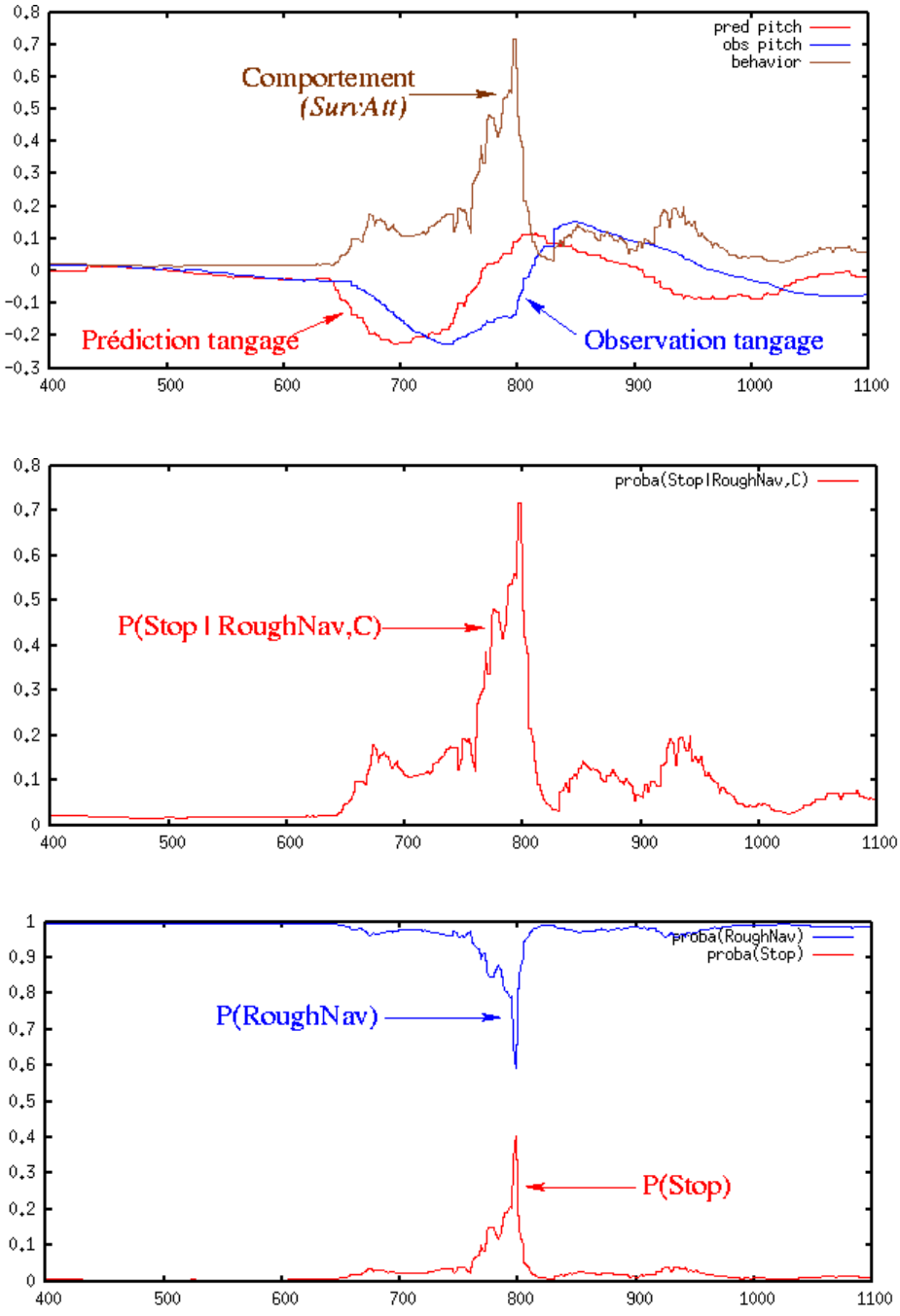


FIG. 6.7 – **Moniteur *SurvAtt* :**
Probabilités d'application des modes et probabilité de transition de *RoughNav* vers *Stop* lors d'une montée de trottoir similaire à la précédente mais avec *SurvAtt* générant un indicateur de mauvais comportement moins grand. Les informations de contexte et les probabilités initiales sont les mêmes que dans le cas précédent (figure 6.6).

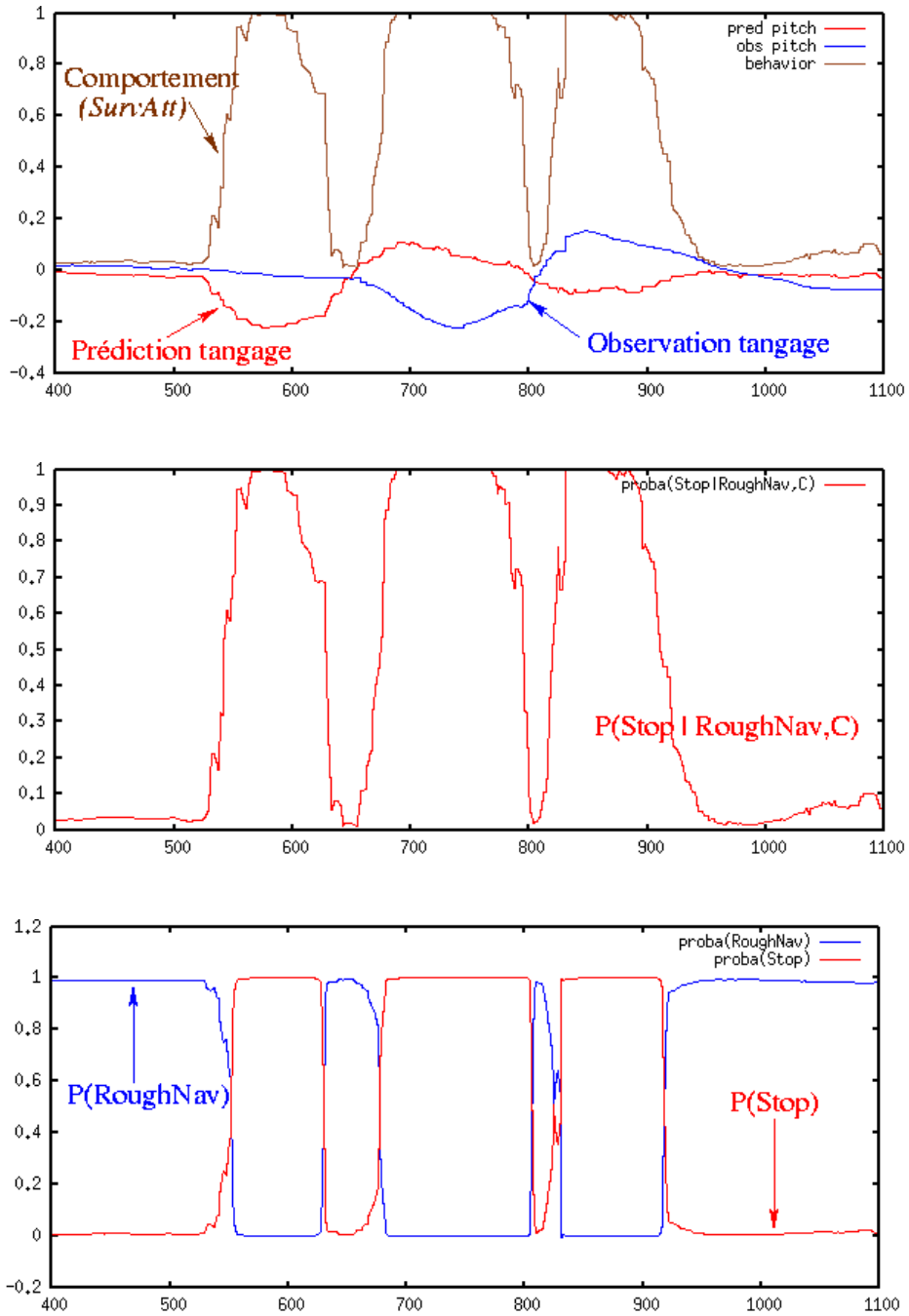


FIG. 6.8 – **Moniteur *SurvAtt*** avec erreur de localisation :
Probabilités d'application des modes et probabilité de transition de *RoughNav* vers *Stop* lors d'une montée de trottoir avec une erreur de localisation (ajoutée manuellement) de 0,5 m en x_{Robot}, direction du déplacement. Les données de contexte et les probabilités initiales sont les mêmes que dans les deux cas précédents. On peut voir que dans cette situation, malgré les données de contexte nettement favorables au mode *RoughNav*, notre système recommande le passage au mode *Stop* pendant presque toute la durée du franchissement du trottoir (qu'il soit "virtuel" ou réel).

moniteurs (on peut par exemple plus facilement ajouter un moniteur supplémentaire), mais n'est aisément réalisable que pour certains types de relations entre ces données :

- Si les moniteurs à combiner estiment le même événement à partir de données indépendantes, une fusion des probabilités sur cet événement obtenues pour chaque moniteur peut être réalisée en appliquant la formule de Bayes ;
- S'il existe un lien causal (une dépendance) entre les événements évalués par les moniteurs à combiner, on peut utiliser une représentation de type Réseau Bayesien Dynamique (DBN) et exploiter les liens entre probabilités conditionnelles ;
- S'il existe un lien logique entre les événements estimés par les moniteurs (un *ou* par exemple), on peut utiliser les propriétés des opérations logiques sur les probabilités, particulièrement si les événements des moniteurs sont *indépendants*.

Combinaison de moniteurs sur Dala

La combinaison choisie est de type *logique*. La transition de *FlatNav* vers *RoughNav* sera ainsi favorisée en cas de détection de terrain accidenté (moniteur *DAT*, présenté en section 5.3) *OU* en cas de glissements repérés par le moniteur *LEM* (section 5.2).

En considérant que les événements de mauvais comportement révélés par le *LEM* (cas de glissement) et le *DAT* sont indépendants, nous exprimons la contribution de comportement des deux moniteurs combinés par une pseudo-probabilité de la disjonction de deux événements indépendants [Peynot et al., 2005] :

$$Q_{FlatNav,RoughNav}(C) = P(Glissement) + P(DAT) - P(Glissement).P(DAT)$$

où $P(Glissement)$ est la probabilité donnée par le *LEM* d'avoir une situation de glissement et $P(DAT)$ la pseudo-probabilité d'avoir un accident de terrain (fournie par le moniteur *DAT*).

Considérons maintenant le cas du mode *RoughNav* : son comportement et les transitions depuis l'état $x_{RoughNav}$ dans la figure 6.4. Nous avons vu que la seule alternative en cas de mauvais comportement de ce mode était le "non-mode" *Stop*. Les données de comportement C seront donc consacrées à la mise à jour de la probabilité de transition conditionnelle $P(RoughNav|Stop, C)$. Dans le mode *RoughNav*, nous sommes en mesure d'évaluer deux types de comportement non satisfaisants (en excluant le moniteur *Locomp*, en perspective) :

- mauvaise prédiction de l'attitude du robot par rapport à ce qui est observé, repérée par le moniteur *SurvAtt* (section 5.4),
- faute de locomotion, repérée par le moniteur *LEM* (état interne s_2 pour ce moniteur, tel que défini dans la section 5.2 86).

Notons les probabilités (ou pseudo-probabilités) de ces événements respectivement $P(SurvAtt)$ et $P(FauteLoco)$. En considérant ces derniers comme indépendants, nous pouvons réaliser une combinaison des deux moniteurs mis en jeu d'une manière similaire à ce qui vient d'être vu, c'est à dire que l'influence de la probabilité de transition par le comportement s'appliquera si *LEM* estime qu'il y a faute de locomotion *ou* si *SurvAtt* détecte un problème de comportement. Ainsi, la probabilité de transition conditionnelle de $x_{RoughNav}$ vers x_{Stop} s'exprimera :

$$P(Stop|RoughNav, C) = P(Stop|RoughNav) + (1 - P(Stop|RoughNav))Q_{RoughNav,Stop}(C)$$

avec :

$$Q_{RoughNav,Stop}(C) = P(SurvAtt) + P(FauteLoco) - P(SurvAtt).P(FauteLoco)$$

Expérimentations

Lors des tests illustrés dans les figures 6.9 et 6.10 les probabilités d'observation sont toujours fixées par un opérateur humain. Ces figures présentent des cas de combinaison d'information de comportement fournies par les moniteurs *DAT* (détection d'accident de terrain à partir des données IMU) et *LEM* (détection de glissement sur information proprioceptives).

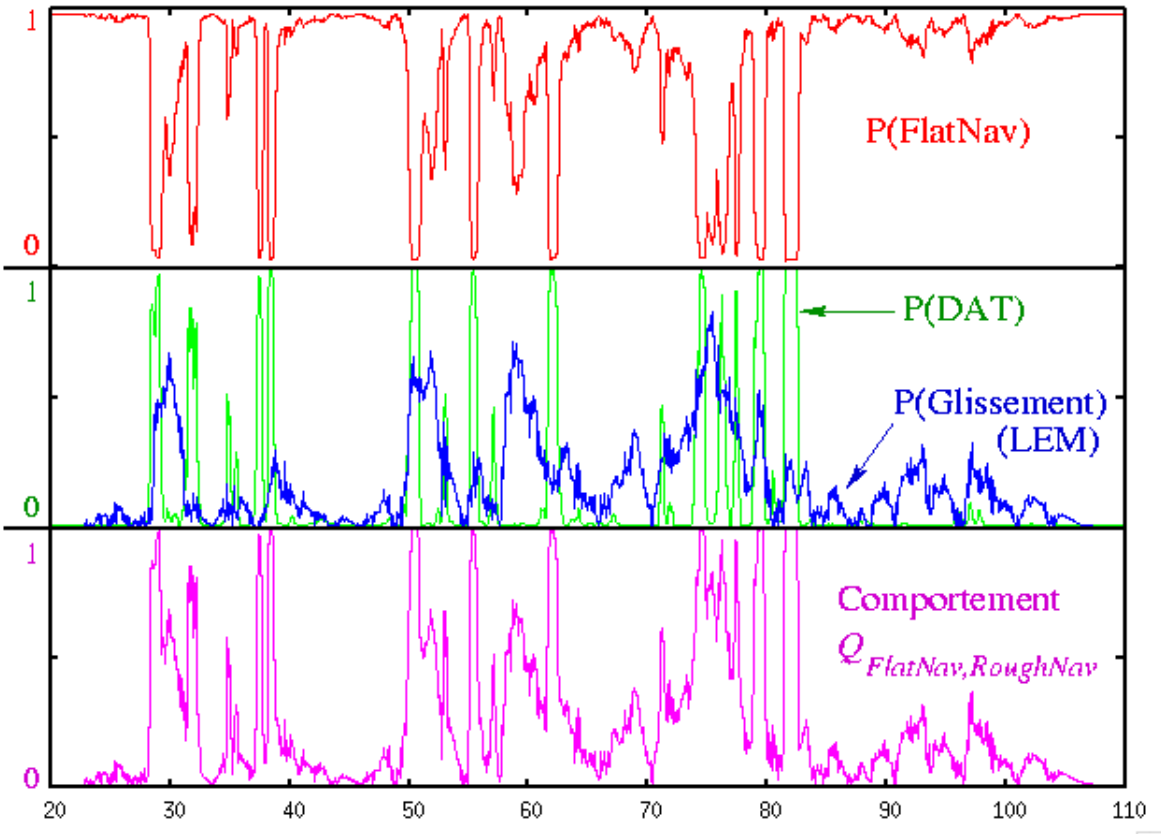


FIG. 6.9 – Exemple de probabilités de mode à appliquer avec données de contexte fixées par un opérateur humain et combinaison des moniteurs *DAT* et *LEM*.
Les probabilités d'observation sont ici fixées à : $p(O|Plat) = p(O|FlatNav) = 0.75$, $p(O|Accidente) = p(O|RoughNav) = 0.20$ et $p(O|Obstacle) = p(O|Stop) = 0.05$. Les probabilités initiales sont : $P(FlatNav) = 0.9$, $P(RoughNav) = 0.09$ et $P(Stop) = 0.01$.
Malgré des données de contexte nettement favorables au mode *FlatNav*, les deux moniteurs combinés contribuent à plusieurs reprises à favoriser plutôt le mode *RoughNav* en repérant des symptômes de terrain accidenté et/ou de mauvaise locomotion (glissements).

6.2.4 Observations issues de la *difmap*

Dans cette partie, les données d'observation sont cette fois fournies par une *difmap* construite par le robot. Sachant la position de ce dernier à l'instant t, on consulte les valeurs des probabilités partielles de contexte dans la *difmap* construite afin de fournir les probabilités $p(O|classe\,de\,terrain)$, qui varient donc au cours du temps dans le test.

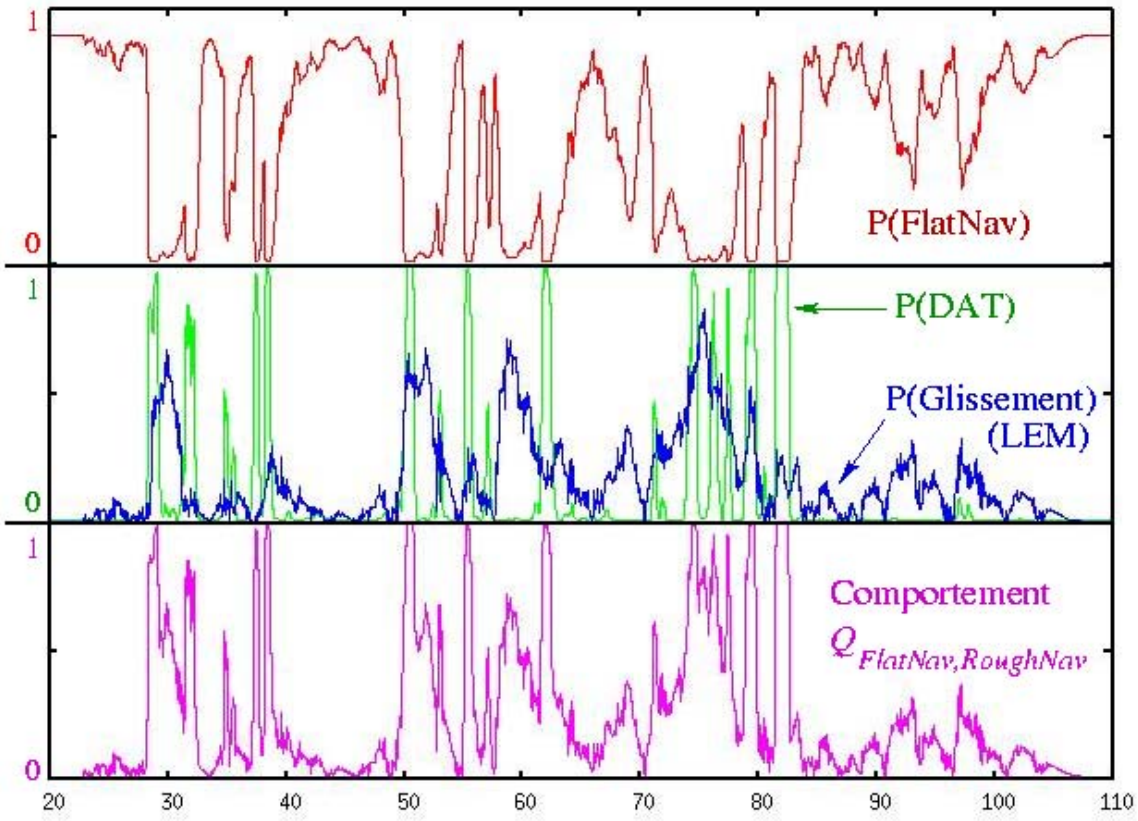


FIG. 6.10 – Un autre exemple dans la même situation que la figure précédente mais avec des probabilités d'observation du contexte différentes (toujours fixées par un opérateur humain) : $p(O|Plat) = p(O|FlatNav) = 0.60$, $p(O|Accidente) = p(O|RoughNav) = 0.38$ et $p(O|Obstacle) = p(O|Stop) = 0.02$. Les probabilités initiales sont quant à elles : $P(FlatNav) = 0.8$, $P(RoughNav) = 0.19$ et $P(Stop) = 0.01$.

La situation expérimentale présentée ici correspond une nouvelle fois au cas "canonique" du robot qui se déplace sur un terrain plat avant de rencontrer un trottoir courbe et de devoir le franchir. La figure 6.11 présente tout d'abord le MNT correspondant à cette situation qui a été construit par le robot, avec la trajectoire réalisée, ainsi que la *difmap* découlant de cette même situation.

La figure 6.12 propose ensuite les probabilités d'observation de contexte qui sont lues dans la *difmap* lors du parcours du robot selon la trajectoire qui vient d'être vue et les estimations de modes à appliquer si l'on ne prend en compte que ces données de contexte, donc sans aucune donnée comportement (les moniteurs étant inactifs, les probabilités de transition sont uniquement celles prédéfinies *a priori*).
Dans cette situation, on peut constater que les données de contexte amènent à favoriser nettement le mode *RoughNav* (par rapport à *FlatNav*) dès que le robot aborde le trottoir modélisé. C'est effectivement une bonne décision si cette modélisation est fidèle à la réalité, ainsi que la localisation du robot. Cependant, sans prise en compte des moniteurs, cet état de fait ne peut être vérifié.

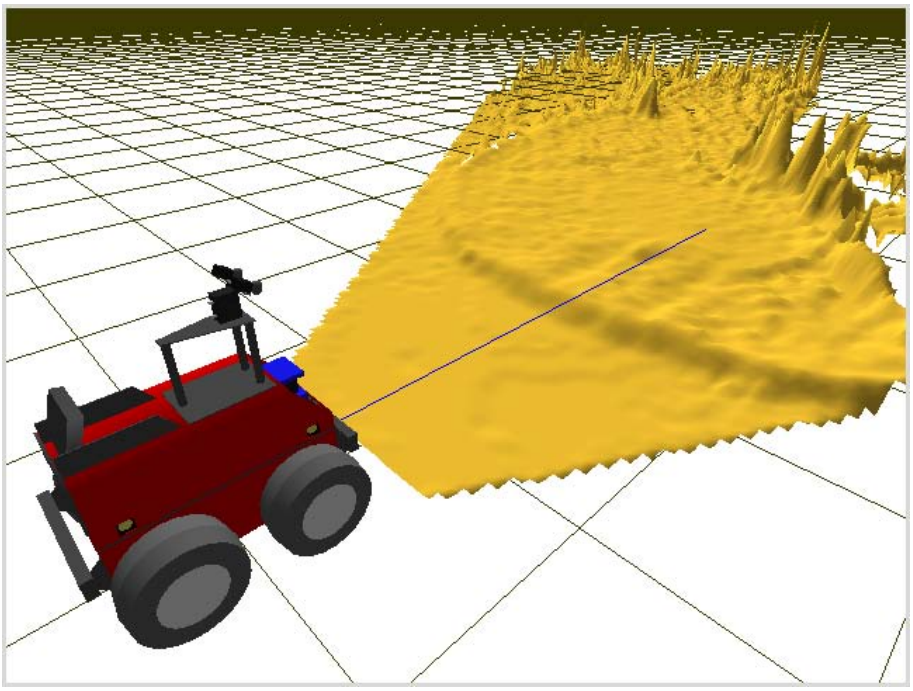
La figure 6.13 propose le même exemple avec prise en compte des données de comportement fournies par le moniteur *SurvAtt*, permettant donc cette fois d'aller dans le sens d'une validation de la modélisation du trottoir, et ainsi de confirmer la préférence de mode initiée par l'observation du contexte.

6.3 Discussion

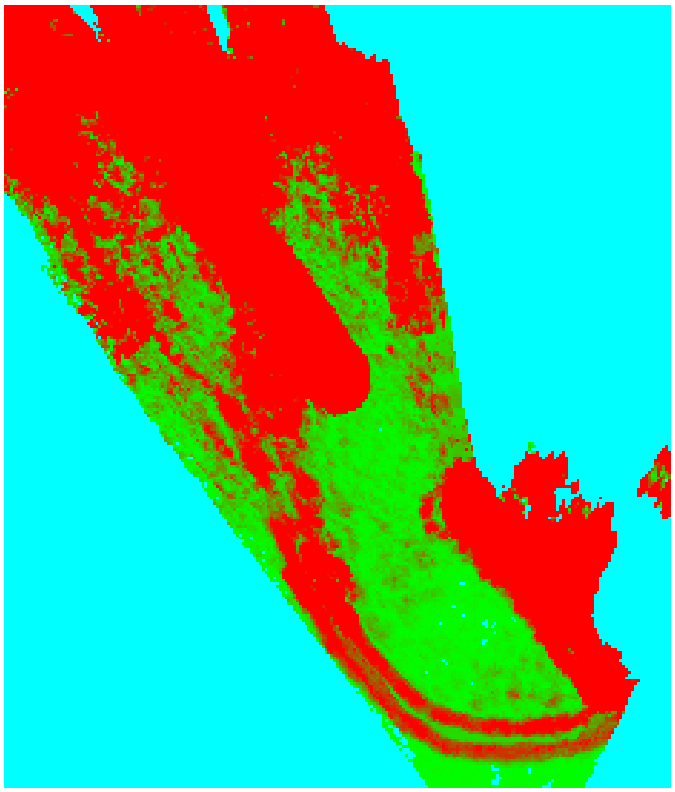
6.3.1 La sélection du mode effectivement appliqué

Nous l'avons vu, le système présenté dans ces travaux propose une estimation du mode de déplacement à appliquer en fonction des observations et du comportement constaté. Il fournit donc des éléments fort utiles pour la *décision*, à savoir quel mode utiliser effectivement, voire des "demandes" de changement de mode. Il reste alors à effectuer en pratique cette décision et "entériner" cette sélection du mode. Pour cela, il convient d'adopter une stratégie plus globale prenant en compte les indications de la méthode d'estimation de modes à appliquer, qui est plutôt *réactive*. Il faut ainsi veiller en particulier à ne pas effectuer de changements trop fréquents ou entrer dans des situations à tendance "instable", voir section 2.3.5 page 42.
Un premier moyen simple d'éviter ce problème de changements de modes trop fréquents est d'appliquer un relais hystérésis (assorti d'une éventuelle temporisation) aux probabilités d'application de mode plutôt que de choisir directement le mode ayant la probabilité d'application la plus élevée. On peut également ajouter une fonction de "coût" de changement de mode, en fonction de la difficulté que peut représenter ce changement (besoin d'arrêter le robot, d'effectuer des initialisations potentiellement longues, etc...).
Pour une stratégie plus globale des changements de mode dans l'espace, nous envisageons d'utiliser une adaptation de la méthode de navigation déjà évoquée dans la section 4.5, issue de [Gancet et al., 2003].

Outre des considérations générales sur les conditions requises ou souhaitées pour des changements de mode effectifs, certains éléments spécifiques au système sont à prendre en considération. Ils sont dus en particulier aux restrictions d'utilisation des modes exploités (conditions nécessaires sur le contexte).
Reprenons l'exemple du système multi-modes de navigation de Dala. Nous avons vu que le mode *FlatNav* nécessitait un contexte de *terrain plat* pour pouvoir être utilisable. Par contre, le mode *RoughNav*, plus lent, est dédié aux terrains accidentés mais peut également être utilisé en terrain

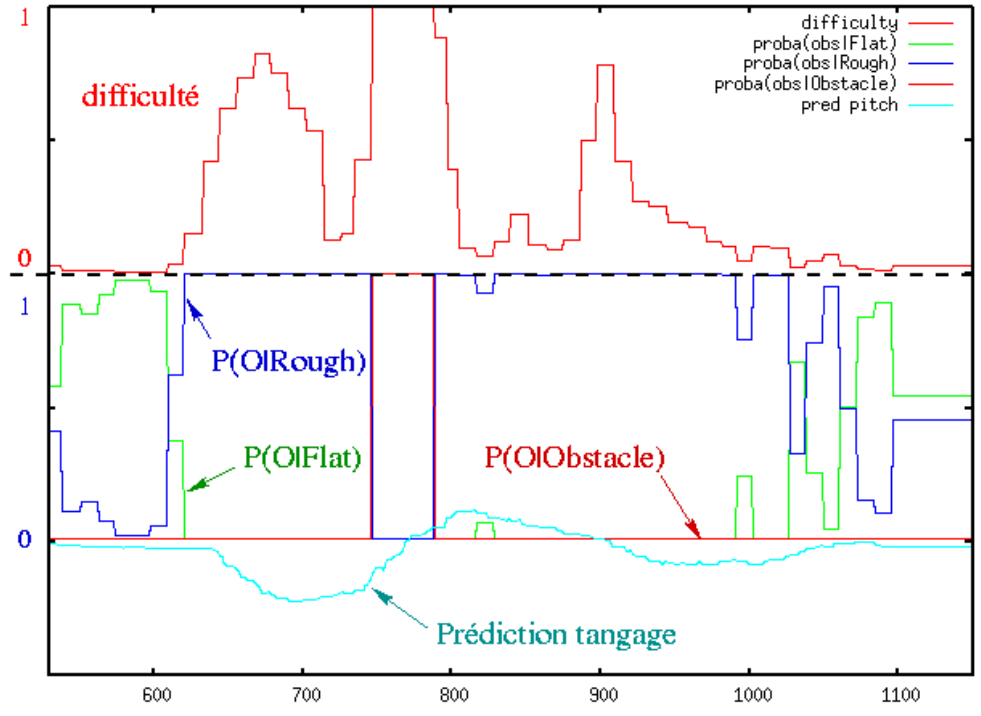


(a) Le MNT et la trajectoire du robot (trait bleu)

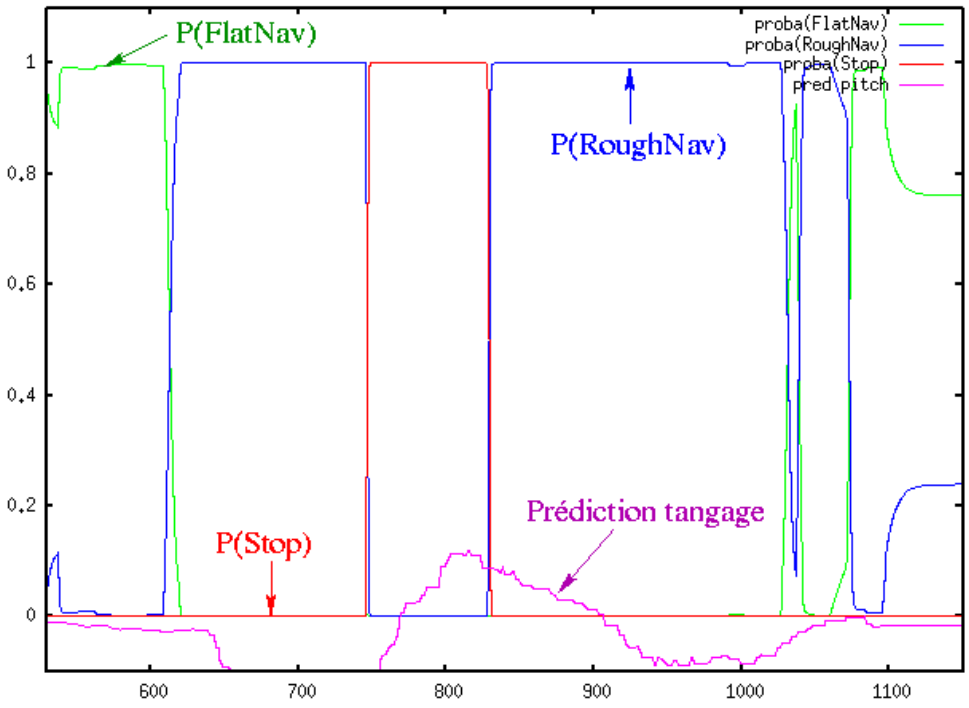


(b) La *difmap* correspondante

FIG. 6.11 – MNT et *difmap* correspondant à la situation expérimentale présentée ici : franchissement d'un trottoir courbe. Le sol précédant le trottoir est plat et couvert de bitume, alors que le sol derrière celui-ci est composé d'herbe et de terre.



(a) Les probabilités d'observation du contexte issues de la *difmap*



(b) Les probabilités d'application de mode

FIG. 6.12 – Illustration des probabilités d'observation ($diff$) sachant les classes de terrain (et les modes de déplacement associés), lues dans la *difmap* construites par le robot ; et les probabilités d'application des modes, sans prise en compte de données de comportement (absence de moniteurs).

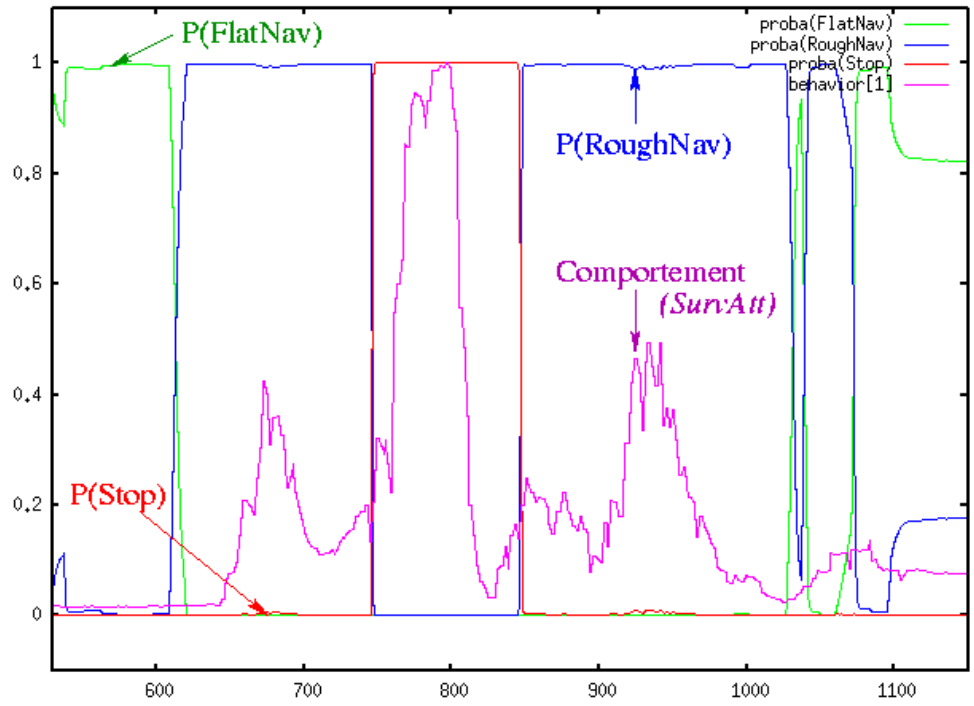


FIG. 6.13 – Même test que précédemment mais avec prise en compte des données de comportement. Ces dernières vont ici plutôt dans le sens des observations du contexte.

plat. C'est pourquoi, du fait de ces restrictions de contexte, il conviendra d'éviter d'ajouter un "coût" supplémentaire ou une temporisation à la transition de $x_{FlatNav}$ vers $x_{RoughNav}$. En effet, si *FlatNav* ne se comporte pas bien, il faut au contraire ne pas hésiter à lui préférer le mode *RoughNav*, plus générique et finalement plus "sûr". Ceci permet une stratégie conservatrice.
De même, dans le cadre d'une telle stratégie, on peut choisir d'effectuer la transition vers le mode *Stop* dès que notre système d'estimation le signifie, c'est à dire en particulier en cas de problème de comportement de *RoughNav*.

6.3.2 Les premiers résultats

Les premiers résultats que nous obtenons avec cette méthode d'estimation du mode de déplacement à appliquer (et avec une réalisation pour des modes de locomotion et surtout une autre plus complète pour des modes de navigation) sont encourageants. Toutefois, sa conception et son développement étant encore assez récents, nous manquons certainement de recul et de retour d'expérience afin de pouvoir la valider significativement.

Pour prouver tout l'intérêt et l'apport d'un tel système, il conviendrait d'effectuer des tests de missions (trajets) de moyenne voire longue durée, en utilisant notre système de sélection de mode associé à un superviseur responsable du choix et des changements à effectuer, puis de réaliser ces mêmes missions sans ce système (donc avec un mode de navigation unique), ou avec uniquement les données de contexte. Nous pourrions alors comparer les performances dans ces différents cas, suivant des critères de rapidité de réalisation de la mission (temps nécessaire pour atteindre les objectifs fixés), d'efficacité (nombre d'échecs du déplacement, énergie dépensée...), voire de sécurité. Cela fait partie des perspectives à court terme de ces travaux.

6.3.3 Démarche de construction de cette méthode

Cette méthode s'est construite progressivement, partant d'études de contrôle "bas niveau" du robot pour remonter vers un niveau plus décisionnel. En effet, tout est venu du besoin de données de *comportement* du robot, pouvant traduire en particulier l'efficacité des actions qu'il entreprend. Ce besoin est régulièrement exprimé par nos collègues s'intéressant à la planification ou la supervision des engins autonomes. Pour pouvoir être correctement intégrés au sein d'un robot et notamment prendre les bonnes décisions, les travaux concernant ces deux derniers domaines doivent considérer les réalités et détails de la couche fonctionnelle. C'est précisément ce que nous avons souhaité faire : rester proche de la couche fonctionnelle et du bas niveau (si l'on doit prendre des décisions, qu'elles soient simples mais autant que possible bien en accord avec ce qui se passe dans le "bas-niveau").
L'avantage de cette démarche est que nous pouvons obtenir des données sur le comportement du robot qui sont proches de la réalité des choses. L'inconvénient est le niveau d'*expertise* qu'il faut avoir afin d'obtenir ces résultats et finalement en déduire de bonnes décisions. .

6.3.4 De l'expertise nécessaire

En plus de la conception des moniteurs plus ou moins "proches des modes" (si possible plus que moins), il nous faut ajuster les paramètres suivants, relevant de l'expertise du concepteur ou de l'utilisateur :

- les probabilités initiales des modes. Elles sont toutefois assez vite "oubliées" dans le déroulement pratique (c'est à dire que leur influence devient vite négligeable au cours du temps),

- la partie "a priori" des probabilités de transition (valeurs fixées par expérience et en fonction du "comportement" global souhaité pour la sélection de modes),
- les paramètres de "sensibilité" des moniteurs (en particulier, paramètres des fonctions de mise en forme pour l'obtention des pseudo-probabilités de comportement),
- le paramétrage du modèle d'observation (pour l'obtention des probabilités d'observation du contexte à partir des données d'observation elles-mêmes).

La mise au point d'un tel système pouvant estimer en ligne quel est le bon mode de déplacement à appliquer demande donc un niveau d'expertise assez conséquent. Nous considérons toutefois que ce niveau d'expertise est utile, voire nécessaire à une bonne prise de décision.

6.3.5 La supervision

Il est primordial que notre système s'intègre judicieusement dans une architecture robotique (l'architecture LAAS, présentée en annexe C.1, en est un exemple notable) pour bien "collaborer", en particulier, avec le superviseur et l'éventuel planificateur du robot, normalement chargés de prendre les décisions de "haut-niveau". Dans ce cadre, nous considérons deux principaux types de collaboration :

- Si le superviseur (et/ou le planificateur) ne décide et ne manipule que des actions assez "haut-niveau", telles que : *aller en (x,y)* ou *prendre une image en cette position*, alors un "mini-superviseur" plus "bas-niveau" pourrait être consacré au choix du bon mode de déplacement directement à partir des probabilités générées par notre système d'estimation, ainsi qu'à l'application effective de ce mode choisi (organisation *hiérarchique*).
- Si par contre le superviseur s'occupe lui-même de gérer toutes les actions du robot, y compris les multiples actions composant la réalisation d'un mode de déplacement (par exemple, pour le mode *RoughNav* : *prendre une image de stéréo-vision*, *mettre à jour le MNT*, *décider d'une trajectoire* etc..., voir l'annexe C), le choix effectif du mode doit être laissé au système de supervision. Néanmoins, notre système d'estimation de mode pourra fournir les informations primordiales à la réalisation de ce choix (sous forme de probabilités partielles ou en signifiant directement le mode recommandé, correspondant à celui qui a la probabilité d'application la plus grande).

Conclusions et perspectives

1 Contributions

Le déplacement entièrement autonome d'un robot mobile en environnements naturels est un problème encore loin d'être résolu. Il nécessite la mise en oeuvre de fonctionnalités permettant de réaliser le cycle perception/décision/action, que nous avons distinguées en deux catégories : *navigation* (perception et décision sur le mouvement à réaliser) et *locomotion* (réalisation du mouvement). Pour pouvoir faire face à la grande diversité de situations que le robot peut rencontrer en environnement naturel, il peut être primordial de disposer de plusieurs types de ces fonctionnalités, constituant autant de *modes de déplacement* possibles. En effet, de nombreuses réalisations de ces derniers ont été proposées dans la littérature ces dernières années mais aucun ne peut prétendre permettre d'exécuter un déplacement autonome en toute situation.
Par conséquent, il semble judicieux de doter un robot mobile d'extérieur de plusieurs modes de déplacement complémentaires. Dès lors, ce dernier doit également disposer de moyens de choisir en ligne le mode qu'il doit appliquer.

Dans ce cadre, cette thèse a proposé une mise en oeuvre d'un système d'estimation du mode de déplacement à appliquer. Cette sélection est réalisée à partir de deux types de données :

- une observation du *contexte* pour déterminer dans quel type de situation le robot doit réaliser son déplacement. Il s'agit principalement de reconnaître les types de contexte auxquels les modes de déplacement disponibles sont dédiés (terrain plat, accidenté...), à partir de moyens de perception extéroceptive comme la stéréo-vision ;
- une surveillance du *comportement* du mode courant, effectuée par des *moniteurs*, et qui influence les transitions vers d'autres modes lorsque le comportement du mode actuel est jugé non satisfaisant.

Ce manuscrit a donc proposé les éléments suivants.

- Un formalisme probabiliste d'**estimation du mode de déplacement à appliquer** a été mis en place. Il est construit sur un HMM.
- Quelques **modes de navigation et de locomotion** que nous avons exploités dans ces travaux ont été présentés. En particulier, un mode de navigation destiné aux terrains accidentés et issu de travaux antérieurs au laboratoire (*RoughNav*) a été adapté à un autre type de plate-forme et complété de données utiles à sa surveillance, et une commande en vitesse pour des robots à roues non directionnelles a été développée pour un mode de locomotion par roulement. Des systèmes multi-modes de déplacement destinés à trois types de plates-formes différents ont alors été introduits.
- Une méthode de **représentation qualitative du terrain** capable de fournir les données de contexte nécessaires au formalisme de sélection de mode a été proposée. Elle repose sur l'évaluation d'une *difficulté* calculée après placement de la structure du robot sur un

modèle numérique du terrain.

– Pour effectuer la surveillance des modes de déplacement utilisés et fournir les données de comportement au système de sélection de modes, des **moniteurs** ont été développés. Le premier effectue une évaluation de l'efficacité de la locomotion par roulement grâce à un processus de classification bayesienne exploitant un apprentissage supervisé préalable. Le deuxième surveille la validité de l'hypothèse de terrain plat, nécessaire à l'application du mode de navigation *FlatNav*, à partir des mesures de capteurs proprioceptifs. Le troisième vérifie la validité de la fonction de placement qui est au coeur du mode de navigation *RoughNav*, en comparant les prédictions de celle-ci avec les observations réalisées en ligne. D'autres moniteurs ont également été proposés.

Les premières expérimentations intégrées qui ont été réalisées tendent à démontrer l'intérêt du changement pour des modes "alternatifs", en particulier lors de la mise en défaut du mode actif. La validation complète de cette approche nécessiterait toutefois des campagnes d'expérimentations plus poussées, afin de couvrir la plus grande diversité de situations possible, et sur des missions à long terme.

2 Discussion

Afin d'obtenir le comportement souhaité, un certain nombre de paramètres doivent être ajustés à différents niveaux, depuis le HMM d'estimation du mode à appliquer (probabilités de transitions *a priori*) jusqu'aux moniteurs (paramètres de la "mise en forme" pour obtenir des pseudo-probabilités de comportement, méthode de combinaison). L'avantage de ces ajustements est de permettre à l'utilisateur et au concepteur d'avoir une certaine maîtrise du comportement global souhaité : par exemple, il peut faire en sorte que son système soit plutôt "paranoïaque", changeant de mode à la moindre suspicion de mauvais comportement, ou au contraire plus conservatif. Toutefois, un bon ajustement de ces paramètres est en pratique délicat à réaliser et reste très dépendant du robot et des modes et moniteurs utilisés.

La production des données des moniteurs est en soi un problème délicat. Dans certains cas (voir le *LEM*, section 5.2) on peut produire des probabilités grâce à un apprentissage préalable par exemple. Toutefois, un tel apprentissage n'est pas toujours réalisable, ou peut tout du moins être très coûteux. Nous sommes ainsi souvent amenés à nous "contenter" de ce que nous appelons des *pseudo-probabilités*, qui sont obtenues par des méthodes *ad-hoc*.

L'un des éléments délicats de ce formalisme est la combinaison des données des moniteurs. Nous avons vu en effet qu'il peut y avoir plusieurs moniteurs surveillant le comportement d'un mode donné et que nous avons d'ailleurs tout intérêt à les multiplier. Il faut alors combiner leurs données sur le comportement pour leur prise en compte dans les transitions du système de sélection. Nous avons pu voir qu'il y avait alors différentes manières d'effectuer cette combinaison, mais qu'il pouvait être difficile d'utiliser le même type de méthode pour tous, tant les moniteurs sont susceptibles de révéler des événements différents.

Notre formalisme repose sur les *probabilités additives* pour prendre en compte les incertitudes. Ce choix a été réalisé pour diverses raisons, parmi lesquelles la possibilité de profiter de travaux antérieurs exploitant ce même formalisme, tels que la classification bayesienne de terrain pour la construction de cartes qualitatives de traversabilité. Dans certains cas, ce choix peut toutefois être remis en cause. En effet cette théorie manipule des arguments *mixtes*, c'est-à-dire que s'ils

prouvent une proposition dans certains cas, ils prouvent le contraire dans les autres (par opposition avec les arguments *purs* qui, s'ils prouvent une proposition dans certains cas, ne disent rien sur les autres) [Bloch, 1996]. Or, ce caractère *mixte* des arguments n'est pas garanti dans tous les composants de notre formalisme.
En effet, par exemple, le rôle des moniteurs combinés est d'évaluer le comportement d'un mode en fonctionnement. Or s'ils permettent bien de révéler un problème de comportement identifiable par les signatures étudiées, un tel événement peut également se produire suite à d'autres causes non identifiables par les seuls moniteurs implémentés. Par conséquent, le fait que nos moniteurs n'identifient pas un mauvais comportement ne signifie pas nécessairement la proposition contraire, c'est-à-dire que le comportement est bon.
D'autre part, les moniteurs ne peuvent renseigner que sur le comportement du mode *actif* à un moment donné ; en effet, dans la plupart des cas nous ne pouvons savoir véritablement comment se comporterait un autre mode s'il était actif. De ce fait, l'utilisation d'un formalisme manipulant des arguments *purs*, tel que la théorie de l'évidence de Dempster-Shafer, pourrait être envisagée.

3 Perspectives

Comme nous l'avons vu, si nous avons pu tester les différents moniteurs développés, les expérimentations avec le système complet intégré qui ont pu être réalisées ne sont encore que préliminaires. Il nous faut en réaliser bien d'autres, et au cours de missions de plus longue durée. Ceci constitue naturellement notre prochain objectif, afin de compléter la validation de notre approche.

Nous travaillons actuellement à la réalisation d'une démonstration de collaboration d'engins autonomes terrestre et aérien, impliquant le robot Dala, équipé du système multi-modes de navigation qui a été proposé, et le ballon dirigeable du LAAS-CNRS, Karma. Dans le scénario prévu, une des missions de Karma sera de fournir des informations initiales de *contexte* terrain [Bosch et al., 2006], qui seront combinées avec celles collectées à bord de Dala lui-même par notre méthode.

Le nombre de modes de navigation et de locomotion effectivement exploités dans nos développements est encore faible pour montrer pleinement l'intérêt d'une telle approche multi-modes en environnement naturel. À ce titre, l'une des perspectives à court terme est l'utilisation de modes additionnels et donc également la mise en oeuvre de moniteurs associés.
Concernant les modes de locomotion, cela passe tout d'abord par l'intégration du système proposé dans ces travaux sur un robot à châssis complexe, muni de mobilités internes, si possible actives. Dans cette perspective, une première étude concernant plusieurs modes de locomotion et quelques moniteurs préliminaires a été menée avec le robot hybride roue-patte Hylos 2 du LRP, dans le cadre du projet R2M (voir annexe B). Ces travaux n'ont malheureusement pas encore pu aboutir à une intégration expérimentale. Nous espérons donc pouvoir poursuivre des développements basés sur cette plate-forme dans un futur proche.
Concernant les modes de navigation, une perspective à court terme est l'ajout effectif du mode *suivi de chemin* au système multi-modes de navigation de Dala (voir section 3.4.2 page 69).

Le développement de moniteurs complémentaires, y compris pour les modes déjà exploités, constitue une autre perspective importante de ce travail. En effet, plus nous pourrons disposer de moniteurs plus nous aurons d'informations utiles sur le comportement des modes, élément

essentiel à l'efficacité de notre système.

Une perspective à plus long terme concerne l'étude de la réalisation d'un système de sélection de modes de déplacement similaire mais reposant sur un autre type de formalisme (la théorie de Dempster-Shafer en est un exemple), voire l'association de plusieurs formalismes différents, afin d'effectuer des comparaisons de performance, robustesse et modularité, particulièrement au niveau des moniteurs.

Le problème que nous avons abordé est d'une importance fondamentale, car il permet d'assurer une exécution correcte et sûre de la navigation, et dans ce sens sa résolution contribue grandement à l'autonomie. Il est pourtant à notre connaissance encore très peu abordé dans la littérature, particulièrement pour les applications en environnements naturels, là où le besoin d'une autonomie complète est souvent le plus manifeste et où les situations peuvent être les plus variées. Nous avons proposé une première approche que nous pensons pertinente, et qui a donné des premiers résultats encourageants. Il nous paraît important de continuer dans cette voie.

Annexe A

Méthodes de localisation

A.1 Odométrie 3D

L'odométrie est certainement la méthode de localisation la plus couramment employée pour les robots disposant d'une structure de locomotion à roues. Son principe consiste à déduire une position, de façon incrémentale, à partir de la vitesse et de la géométrie des roues. La mise en œuvre de cette méthode est des plus simples et ne nécessite qu'une puissance de calcul très limitée.

L'odométrie fournit la mesure de la longueur Δs de l'arc de cercle parcouru, calculée à partir des vitesses des roues centrales gauche (ω_l) et droite (ω_r) (figure A.1) :

$$\begin{aligned} \Delta s &= \Delta t\, R\, (\omega_r + \omega_l)/2 \\ \Delta\theta &= \Delta t\, R\, (\omega_r - \omega_l)/e \end{aligned} \tag{A.1}$$

où R est le rayon des roues, e l'entraxe[1] et Δt la période d'échantillonnage des signaux.

Les déplacements élémentaires en trois dimensions $(\Delta x, \Delta y, \Delta z)$ s'expriment en fonction de l'abscisse curviligne Δs, de la variation d'angle de cap $\Delta\theta$ et de l'inclinaison $\Delta\phi$ du plan dans lequel le robot évolue grâce aux équations de l'odométrie 3D (figure A.2) [Mallet, 2001] :

$$\begin{aligned} \Delta x &= \Delta s\, cos(\Delta\theta/2)\, cos(\Delta\phi) \\ \Delta y &= \Delta s\, sin(\Delta\theta/2)\, cos(\Delta\phi) \\ \Delta z &= -\Delta s\, sin(\Delta\phi) \end{aligned}$$

Une amélioration notable de cette odométrie peut être apportée avec l'exploitation du gyromètre : la valeur de $\Delta\theta$ n'est plus calculée par l'estimation (A.1) (les glissements génèrent en effet trop d'erreurs) mais est donnée par un gyromètre à fibre optique, précis et ayant une faible dérive. Cette *gyrodométrie* [Borenstein et al., 1996a] est exploitée sur les robots Lama et Dala. Notons que cette odométrie 3D nécessite également une mesure ou une bonne estimation des angles d'attitude du robot.

[1] l'entraxe se définit comme la distance moyenne entre les 2 surfaces de contact avec le sol et dépend, en toute rigueur, de ce dernier. Ceci dit, nous estimerons qu'il est constant, égal à la longueur d'un essieu.

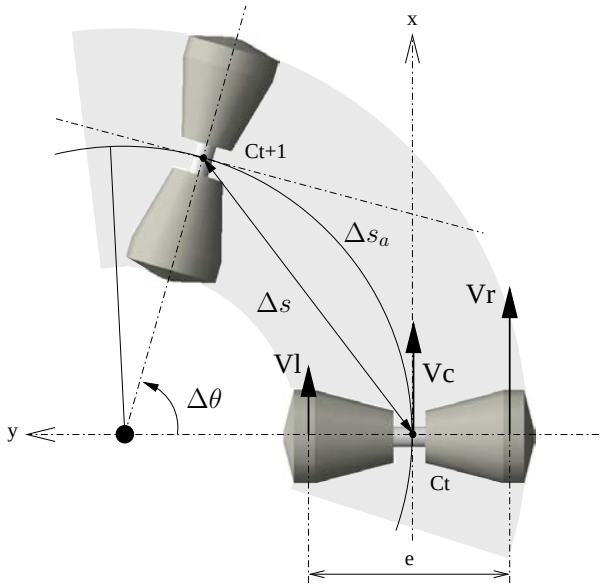


FIG. A.1 – Calcul de la position odométrique, d'après [Mallet, 2001] : connaissant les vitesses de chacune des roues, il est possible de calculer périodiquement un déplacement élémentaire (représenté de façon agrandie ici) que nous pouvons approximer par un arc de cercle de longueur Δs_a et de variation angulaire $\Delta\theta$. La longueur Δs de la corde de cet arc étant petite, elle peut être approximée par la longueur Δs_a de l'arc. Les déplacements élémentaires sont cumulés et la position peut ainsi être exprimée dans un repère global.

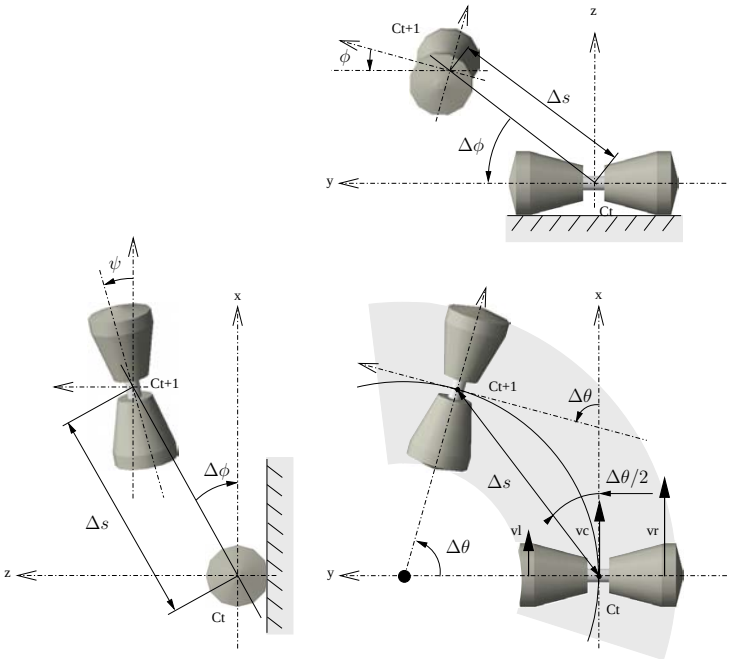


FIG. A.2 – Calcul de la position odométrique 3D, d'après [Mallet, 2001] : le robot évolue dans un plan qui n'est pas horizontal. L'inclinaison moyenne $\Delta\phi$ de ce plan est souvent approximée par la moyenne des angles de site entre deux positions successives. Nous constatons par ailleurs que l'angle de gîte ψ n'influence pas la position de l'essieu.

A.2 Estimation de l'attitude grâce à l'IMU

Lama dispose d'inclinomètres fournissant une mesure directe des angles d'attitude du robot (roulis et tangage). Dala ne dispose pas quant à lui de capteurs de ce type, mais d'une centrale inertielle (IMU) trois axes, composée d'accéléromètres et de gyromètres sur les 3 axes liés au robot. Afin de produire une estimation de position 3D, nous avons besoin d'effectuer une estimation de ces angles d'attitude, qui ne peut être obtenue sur Dala que grâce à l'IMU. Nous avons pour cela implémenté une méthode adaptée de [Metni et al., 2005], réalisant l'estimation non-linéaire d'attitude par la technique du backstepping.
Elle part des constatations suivantes :

- L'intégration des données des gyromètres fournit une bonne prédiction de l'attitude *à court terme* mais l'intégration en boucle ouverte fait vite dériver l'estimation.
- Les mesures des accéléromètres sont bonnes pour estimer la direction du vecteur gravité en "régime permanent", mais sont très bruitées et sensibles aux accélérations parasites.

La méthode propose donc de combiner les deux capteurs, profitant des avantages de chacun, pour fournir une estimation de la direction de la gravité et adapter le biais du gyroscope en temps réel. La direction de la gravité est ainsi :

- prédite par l'intégration des mesures du gyroscope,
- corrigée par les données des accéléromètres.

Posons comme précédemment ϕ l'angle de tangage, ψ celui de roulis et $\boldsymbol{x} = (x_1\, x_2\, x_3)^T$ le vecteur de gravité normalisé. Notons également, en général, $\hat{\boldsymbol{y}}$ l'estimée du vecteur $\boldsymbol{y}$.

La structure de l'observateur du vecteur de gravité est la suivante (voir également figure A.3) :

$$\dot{\hat{\boldsymbol{x}}} = -\int (\boldsymbol{\Omega_m} + \boldsymbol{\omega'}) \wedge \hat{\boldsymbol{x}} \tag{A.2}$$

$$\boldsymbol{\omega'} = (\hat{\boldsymbol{x}} \wedge \boldsymbol{x}) k_m + \hat{\boldsymbol{b}} \tag{A.3}$$

$$\hat{\boldsymbol{b}} = \int k_b (\hat{\boldsymbol{x}} \wedge \boldsymbol{x_m}) \tag{A.4}$$

avec :

- $\boldsymbol{\Omega_m}$: vecteur de rotation mesuré par le gyroscope de l'IMU,
- $\boldsymbol{x_m}$: vecteur des accélérations mesurées par l'IMU (normalisé),
- $\boldsymbol{b}$: biais des gyromètres,
- k_m et k_b : gains, réglés dans notre application respectivement à 1.0 et 0.1.

Connaissant la direction du vecteur gravité, on peut alors facilement déduire les angles d'attitude recherchés. En effet :

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \sin\phi \\ -\cos\phi \sin\psi \\ -\cos\phi \cos\psi \end{pmatrix}$$

D'où nos estimations d'angles :

$$\hat{\phi} = \arcsin \hat{x}_1 \tag{A.5}$$

$$\hat{\psi} = -\arcsin(\hat{x}_2 / \cos\hat{\phi}) \tag{A.6}$$

$$\hat{\psi} = -\arccos(\hat{x}_3 / \cos\hat{\phi}) \tag{A.7}$$

Ces angles peuvent alors être utilisés pour l'odométrie 3D ou pour compléter une position GPS qui ne donne pas les orientations.

Notons que cette méthode ne permet pas d'obtenir directement les variances des estimées de ces angles d'attitude.

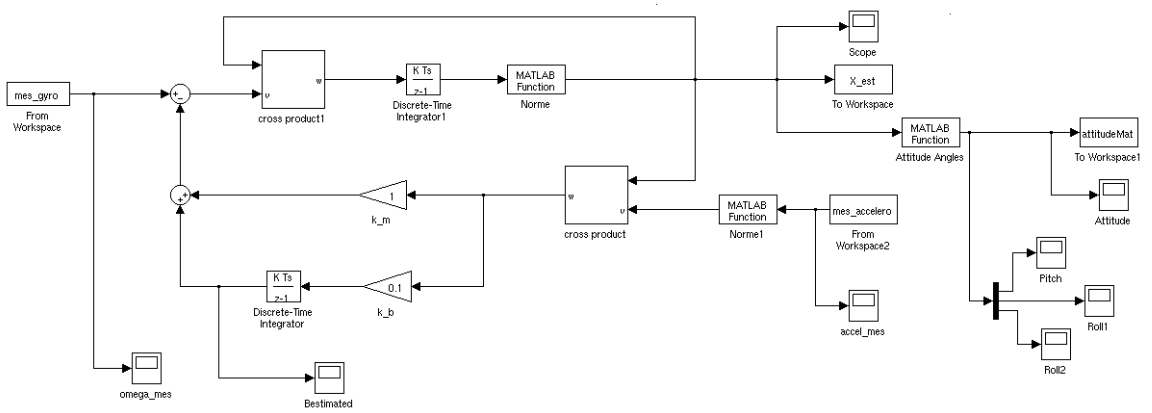


FIG. A.3 – Schéma bloc de l'observateur d'attitude sur Dala, à partir des données de l'IMU

A.3 Estimation visuelle de mouvement (*odométrie optique*)

Cette méthode effectue une estimation des déplacements *relatifs* (d'où son nom d'*odométrie* optique) réalisés entre deux acquisitions de paires d'images consécutives par le banc de stéréovision [Mallet et al., 2000]. Elle repose tout d'abord sur une sélection de *points d'intérêt* (points de Harris), pixels dans l'image possédant des propriétés intéressantes qui permettront de les suivre dans les paires d'images successives. On effectue sur ces points un appariement stéréoscopique (i.e. entre les images droite et gauche), obtenant ainsi, grâce aux paramètres de calibration, autant de points 3D. On procède ensuite à un nouvel appariement entre des points des images acquises à t et à $t+\delta t$ (voir figure A.4), qui permet de déduire une estimation de la transformation 3D complète entre les points aux deux instants successifs, c'est-à-dire le déplacement relatif réalisé pendant le temps δt. Les six paramètres de cette transformation 3D (trois paramètres de translation et trois de rotation) sont fournis avec des écarts-types associés.

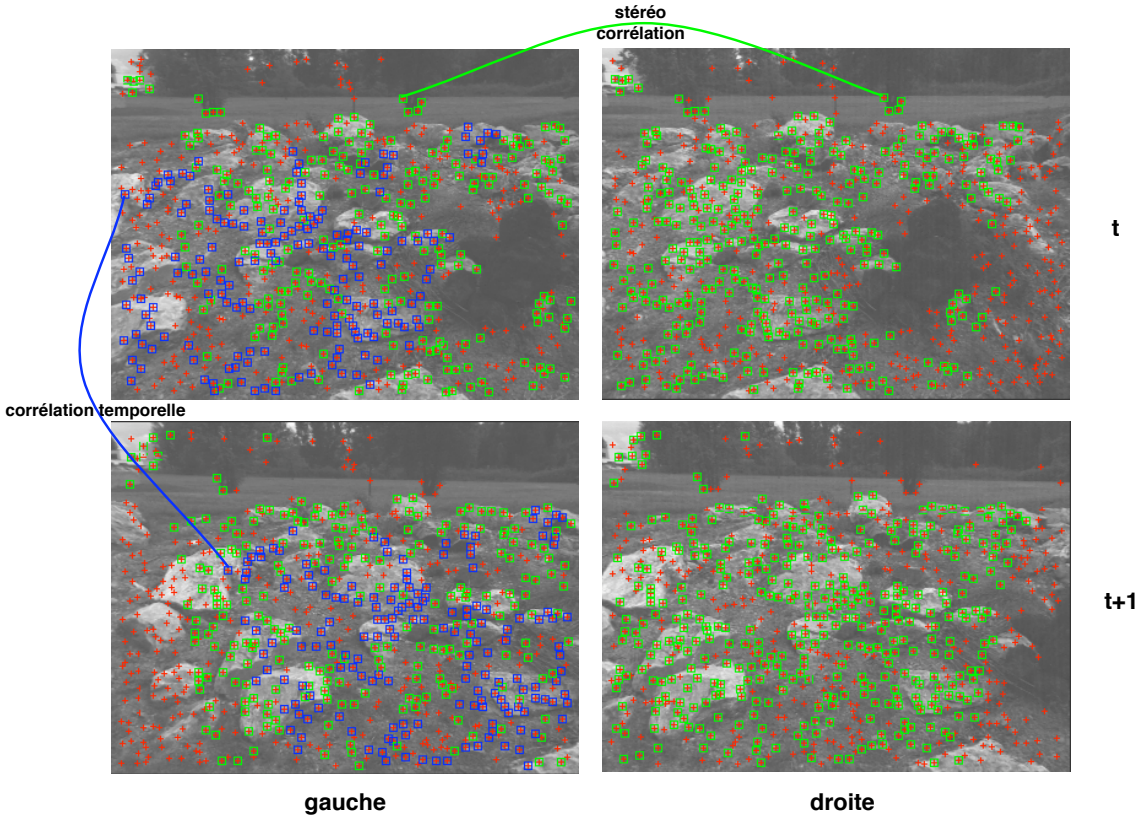


FIG. A.4 – Principe de l'estimation visuelle de mouvement (*odométrie optique*), illustration d'après [Py, 2005]

Annexe B

Hylos 2 et le projet R2M

B.1 Le projet R2M

R2M (*Rover Multi-Modes pour une haute mobilité sur terrain accidenté*) est un projet du programme ROBEA[2] qui a débuté en novembre 2003 et s'est conclu début avril 2006 [Martinet et al., 2006]. Il a concerné les mini-rovers rapides et à fortes capacités de franchissement, et plus particulièrement le développement d'une telle plate-forme, munie de modes de locomotion multiples pour appréhender des situations différentes, pouvant aller de surfaces relativement régulières à des environnements très déstructurés. Les études menées ont concerné :

- la *conception mécanique* du robot, baptisé Hylos 2 car inspiré d'une première version nommée Hylos,
- le développement de *modes de locomotion différents*, permettant au rover de disposer d'une haute mobilité et d'aborder des situations diverses,
- l'intégration d'un module dit de *perception locale augmentée*, sur la base de données géométriques issues de la stéréo-vision et d'informations de texture grâce à une caméra couleur,
- le *monitoring* et la *sélection du mode* de locomotion en ligne.

Ce dernier point ainsi que la partie géométrique de la perception locale augmentée ont été réalisés par le LAAS-CNRS.

Outre notre laboratoire, des équipes du LRP (Laboratoire de Robotique de Paris), du LASMEA-Cemagref de Clermont-Ferrand et du centre DGA d'Angers (ETAS) ont collaboré à ce projet.

B.2 La plate-forme Hylos 2

Hylos 2 est un robot hybride roue-patte dont le châssis a été mis au point par le Laboratoire de Robotique de Paris (LRP).

B.2.1 Châssis

Ce robot dispose d'une structure poly-articulée lui donnant des possibilités de reconfigurabilité : il est constitué de quatre chaînes roue-patte à 4 degrés de liberté. Chaque patte possède deux liaisons pivot d'axes parallèles, actionnées par des systèmes à vérins à vis. Chaque extrémité de patte est équipée d'une roue motrice et directrice.

[2]Programme national ROBotique et Entités Artificielles

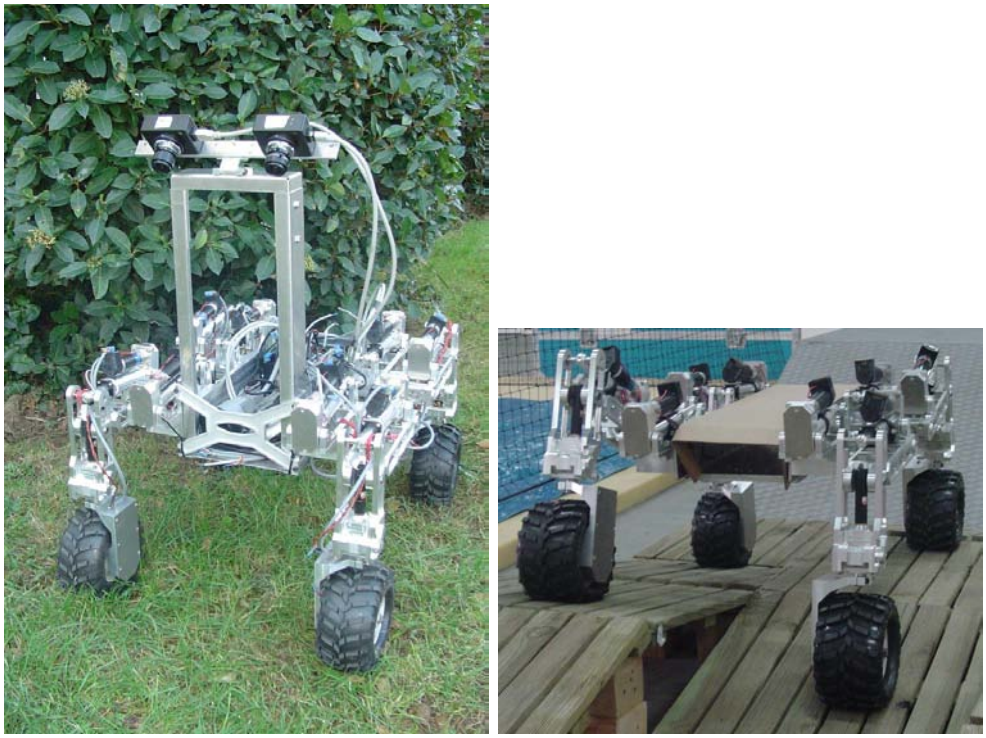


FIG. B.1 – Le robot Hylos 2 du LRP

B.2.2 Capteurs

Hylos 2 est équipé des capteurs suivants :
- une **centrale inertielle** permettant en particulier la mesure des angles de roulis et de tangage du châssis,
- des **codeurs odométriques** sur chaque roue pour mesurer le déplacement réalisé par celles-ci,
- des **codeurs** sur les liaisons des axes, permettant de connaître la configuration complète du robot (positions relatives des éléments de chaque patte),
- un **système de localisation GPS** centimétrique,
- un **banc de stéréo-vision** fourni par le LAAS, avec les fonctions logicielles de traitement associées.

B.3 Modes de locomotion

B.3.1 Présentation des modes

Les quatre modes de locomotion suivants ont été développés au LRP pour permettre à ce robot d'avoir de bonnes capacités de franchissement de terrain accidenté. Ils sont hérités de la première version de ce robot, Hylos :
- Mode 1 : le **roulement pur**, dans lequel la posture du robot est figée, peut permettre des déplacements rapides. Il est plutôt adapté à des sols suffisamment cohésifs et relativement réguliers (voire plats).

- Mode 2 : le **roulement avec reconfiguration** exploite les mobilités internes actives donnant à la structure un nombre relativement important de degrés de liberté. Une loi de commande réalise une régulation des 7 paramètres de posture et des 3 paramètres de localisation/configuration en exploitant un modèle cinématique inverse du robot. Cette commande vise à réaliser une répartition uniforme des forces de contact des roues avec le sol [Grand et al., 2004]. Les mesures exploitées dans cette commande sont celles des capteurs proprioceptifs (paramètres articulaires) et extéroceptifs (angles de roulis et de tangage) pour la correction de posture et l'estimée de localisation pour la correction de trajectoire. Les principaux avantages de ce mode sur le premier sont [Grand, 2004] :
  - l'amélioration de la *marge de stabilité* sur pente (voir B.4), en particulier dans le cas d'un dévers,
  - la quasi-indépendance de la marge de stabilité vis-à-vis du cap.
- Mode 3 : le **péristaltisme** (figure B.2) est semblable à celui exploité sur le robot Lama. Il est d'ailleurs issu des travaux menés conjointement par le LAAS et le LRP pour sa mise au point [Andrade et al., 1998]. Il exploite donc également les mobilités internes actives pour faire "ramper" le robot comme une chenille. Le mouvement s'effectue en deux phases : une première faisant avancer les pattes avant seules, suivies dans une deuxième phase des pattes arrières. Cette allure est symétrique, mais une allure dissymétrique est également possible [Grand, 2004] (figure 3.20 droite). Des rotations peuvent alors être envisagées (contrairement à ce qui était le cas pour Lama, voir 3.3.2), en appliquant une commande de type "skid-steering", c'est à dire en faisant avancer à chaque phase les roues du côté droit et gauche d'une distance différente. Ce mode, bien qu'imposant un déplacement très lent, peut se révéler bien plus efficace (meilleure traction) et plus économe en énergie que les deux modes précédents, sur des sols meubles ou peu cohésifs.
- Mode 4 : **franchissement**. Ce mode est destiné au cas de franchissement d'obstacles frontaux de dimensions proches du diamètre de la roue (rencontrés dans des environnements dits *semi-structurés*).

B.3.2 Système multi-modes de locomotion

Le système multi-modes de locomotion envisagé pour le robot Hylos 2 (voir figure B.3) repose sur les modes *Roulement pur* (RP), *Péristaltisme* (P) et *Roulement avec Reconfiguration* (RR) exploitant les mobilités internes actives du robot pour son déplacement. Les contextes dédiés correspondants sont respectivement les terrains plats cohésifs, les terrains non cohésifs et terrains accidentés. Un robot équipé d'un tel système devrait être armé pour bien négocier la plupart des situations délicates qu'il est susceptible de rencontrer, si toutefois le mode courant est choisi à bon escient.

B.4 Moniteurs envisagés pour la locomotion d'Hylos 2

Dans le cadre du projet R2M, des moniteurs de modes de locomotion pour Hylos 2 ont été étudiés. Ils n'ont malheureusement pas encore pu être intégrés sur le robot car ce dernier est en cours de développement. Nous ne présenterons donc qu'une étude préliminaire sans résultats obtenus sur cette plate-forme. Leur implémentation reste néanmoins en projet une fois que cette dernière sera opérationnelle.

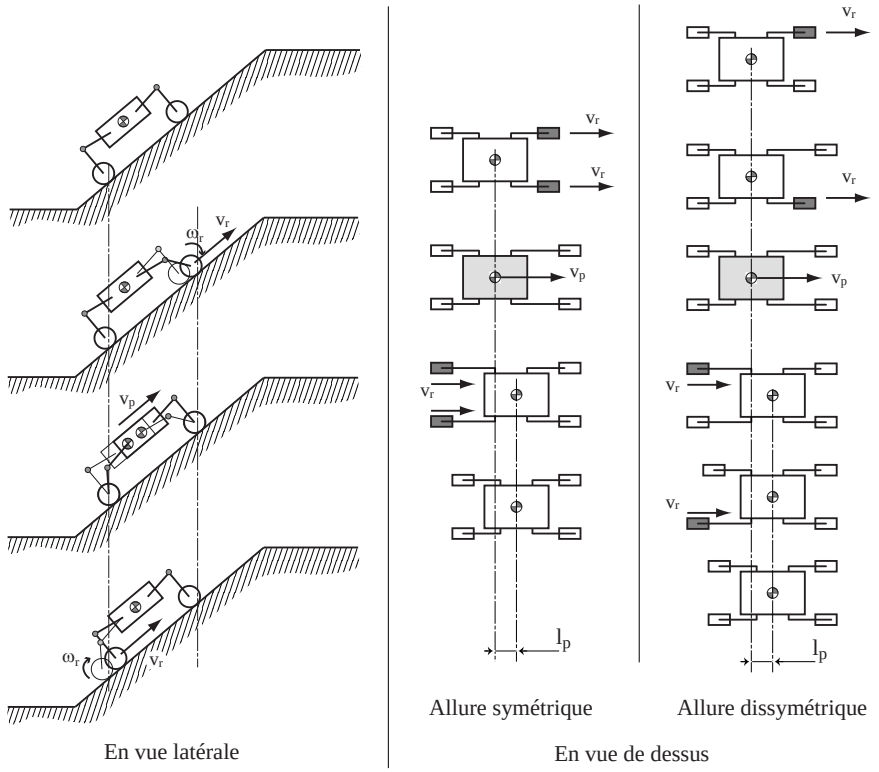


FIG. B.2 – Illustration du mode de locomotion *péristaltique* pour un robot hybride roue-patte comme Hylos, d'après [Grand, 2004].

B.4.1 Marge de stabilité

Les quatre points de contact des roues de Hylos 2 avec le sol (les P_i de la figure B.4) définissent quatre droites de basculement auxquelles peuvent être associés quatre angles de stabilité υ_i. La marge de stabilité totale du véhicule peut être définie comme le minimum des angles : $Ms = \min(\upsilon_i)$ [Grand, 2004]. C'est une métrique avec un domaine d'évolution borné bien défini : elle varie entre 0 et $\pi/2$ (une valeur de Ms en dehors de $[0, \pi/2]$ signifierait que la plate-forme a déjà basculé). Ainsi, après normalisation et application d'une fonction de forme, nous pouvons obtenir une donnée similaire à une probabilité : celle d'avoir une plate-forme instable.

La variation de Ms pourrait aussi être une information intéressante pour détecter les "tendances" qui vont vers le basculement.

Ce moniteur peut être utilisé dans un mode de roulement simple (sans reconfiguration) ou dans un mode de roulement avec reconfigurations destiné aux terrains accidentés.

- Dans le premier cas, le rover est censé se déplacer sur des terrains pratiquement plats, donc en gardant une marge de stabilité pratiquement maximale. Dès que cela n'est plus le cas, le présent moniteur pourrait inciter à se diriger vers un mode destiné aux terrains accidentés (roulement avec reconfiguration par exemple).
- Dans le deuxième cas, le rapprochement d'une marge de stabilité critique pourrait amener le robot à entrer dans un mode *Stop*, la situation devenant trop dangereuse.

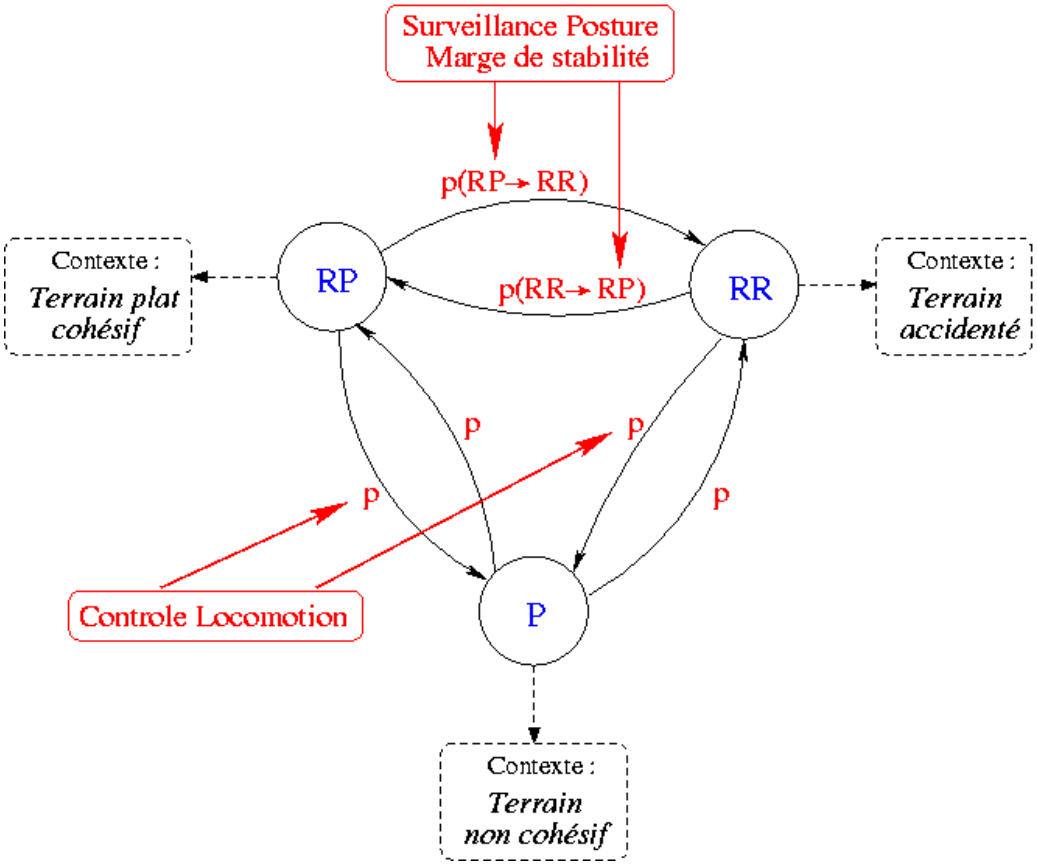


FIG. B.3 – Système multi-modes de locomotion pour Hylos 2

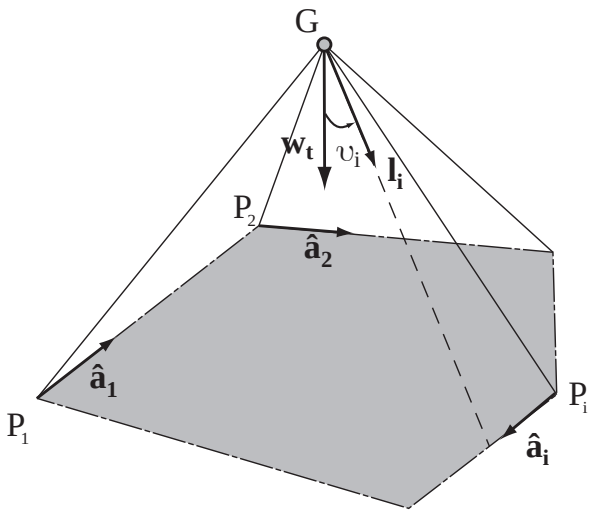


FIG. B.4 – Illustration de la marge de stabilité de la plate-forme Hylos 2, d'après [Grand, 2004].

B.4.2 Surveillance de posture

Les capteurs internes du robot permettent d'avoir à chaque instant une mesure de la configuration de son châssis. La centrale inertielle embarquée permet quant à elle de disposer d'une estimation de son attitude. On peut alors déduire de ces informations la posture du robot, à savoir l'attitude du plateau central d'Hylos 2. L'étude de cette posture dans un mode tel que le roulement pur, dans lequel les articulations actives sont maintenues fixes, peut servir à vérifier l'hypothèse de terrain plat d'une manière similaire au moniteur présenté dans la section 5.3 page 91. Lors du mode *Roulement avec reconfigurations*, la commande de locomotion effectue un asservissement de la posture du robot afin de traverser des terrains très accidentés. Il serait donc tout à fait pertinent de surveiller cette posture afin de vérifier que l'écart avec les prévisions reste acceptable (stabilité de la commande).

Annexe C

Intégration logicielle

L'intégration sur nos robots des différents modes de déplacement et des éléments composant notre système d'estimation de mode à appliquer nécessite l'utilisation de plusieurs outils de développement et l'organisation d'un nombre conséquent de *modules* dans une architecture robotique. Cette annexe vise à présenter brièvement cette intégration sur le robot Dala ainsi que celle de ses modes de navigation.

C.1 L'architecture LAAS

L'architecture *LAAS* (LAAS Architecture for Autonomus Systems, figure C.1) [Alami et al., 1998] est une architecture dédiée aux systèmes autonomes, et en particulier les robots.

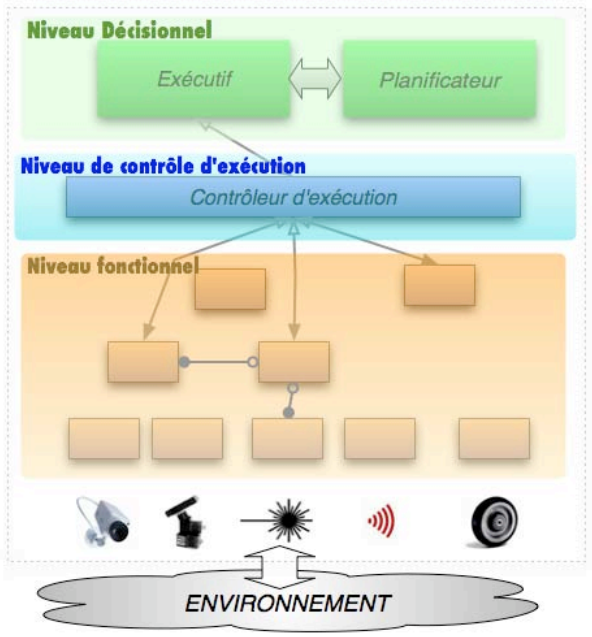


FIG. C.1 – L'architecture *LAAS* (LAAS Architecture for Autonomus Systems)

Elle comporte trois niveaux :
– le *niveau décisionnel*, chargé de la planification et de l'exécution du plan,

- le *niveau de contrôle d'exécution*, chargé de vérifier la validité des requêtes transmises par le niveau décisionnel (d'après des règles pré-établies) et de surveiller les éventuels messages d'erreur renvoyés par le niveau fonctionnel,
- le *niveau fonctionnel*, en communication directe avec les capteurs et actionneurs du robot (et donc avec l'environnement), dont le rôle est de réaliser en pratique les demandes d'action venues du niveau décisionnel. C'est le niveau fonctionnel qui réalise les tâches du cycle de perception/décision/action nécessaires à la navigation et la locomotion du robot.

La couche fonctionnelle est composée de *modules* générés grâce à l'outil `GenoM`, développé au LAAS-CNRS.

C.2 GenoM

`GenoM` (Generator of Modules) [Fleury et al., 1997][Fleury et al., 2005] est un outil de développement pour des robots, permettant de générer des *modules* destinés au niveau fonctionnel de l'architecture LAAS. Il fournit divers services, tels que des moyens de communication entre les modules, des interfaces de communication pour un superviseur ou pour un opérateur par un langage de scripts.
Un module `GenoM` comporte les éléments suivants :

- Des *tâches d'exécution* exécutent, de manière périodique ou non, les fonctions de traitement codées par le concepteur du module. Il peut y en avoir plusieurs fonctionnant en parallèle. Ces fonctions de traitement sont soit permanentes, soit déclenchées par une *requête* adressée au module par l'intermédiaire d'un programme disposant de la bibliothèque d'accès aux services d'un module `GenoM` (voir par exemple `openPRS`, section C.4).
- Une *tâche de contrôle* est chargée de réceptionner les requêtes adressées au module, de vérifier la validité des paramètres, de gérer les conflits ou encore de retourner un bilan sur l'activité en cours.
- Des *posters*, mis à jour régulièrement ou manuellement par les tâches d'exécution, contiennent des informations mises à disposition des autres modules voire d'autres applications telles que celles du niveau décisionnel (voir `openPRS`). Chaque module dispose d'une bibliothèque d'accès aux posters des autres modules actifs. Ainsi, un module de localisation peut par exemple mettre à disposition des autres modules qui en ont besoin la position courante du robot dans un poster.

C.3 Modules fonctionnels de Dala

La figure C.2 illustre l'organisation des modules fonctionnels `GenoM` mis en jeu lors de l'utilisation des modes de navigation *FlatNav* et *RoughNav* sur Dala.

Les modules mis en oeuvre peuvent se ranger en trois catégories :

- les modules impliqués dans la *localisation* du robot,
- les modules constituant le mode de navigation *RoughNav*,
- ceux constituant le mode de navigation *FlatNav*.

C.3.1 Les modules liés à la localisation

- `IMU` (pour Inertial Measurement Unit) est chargé de lire à haute fréquence les mesures de la centrale inertielle (accélérations et vitesses angulaires 3 axes) et d'effectuer l'estimation des angles d'attitude (roulis et tangage, voir section A.2).

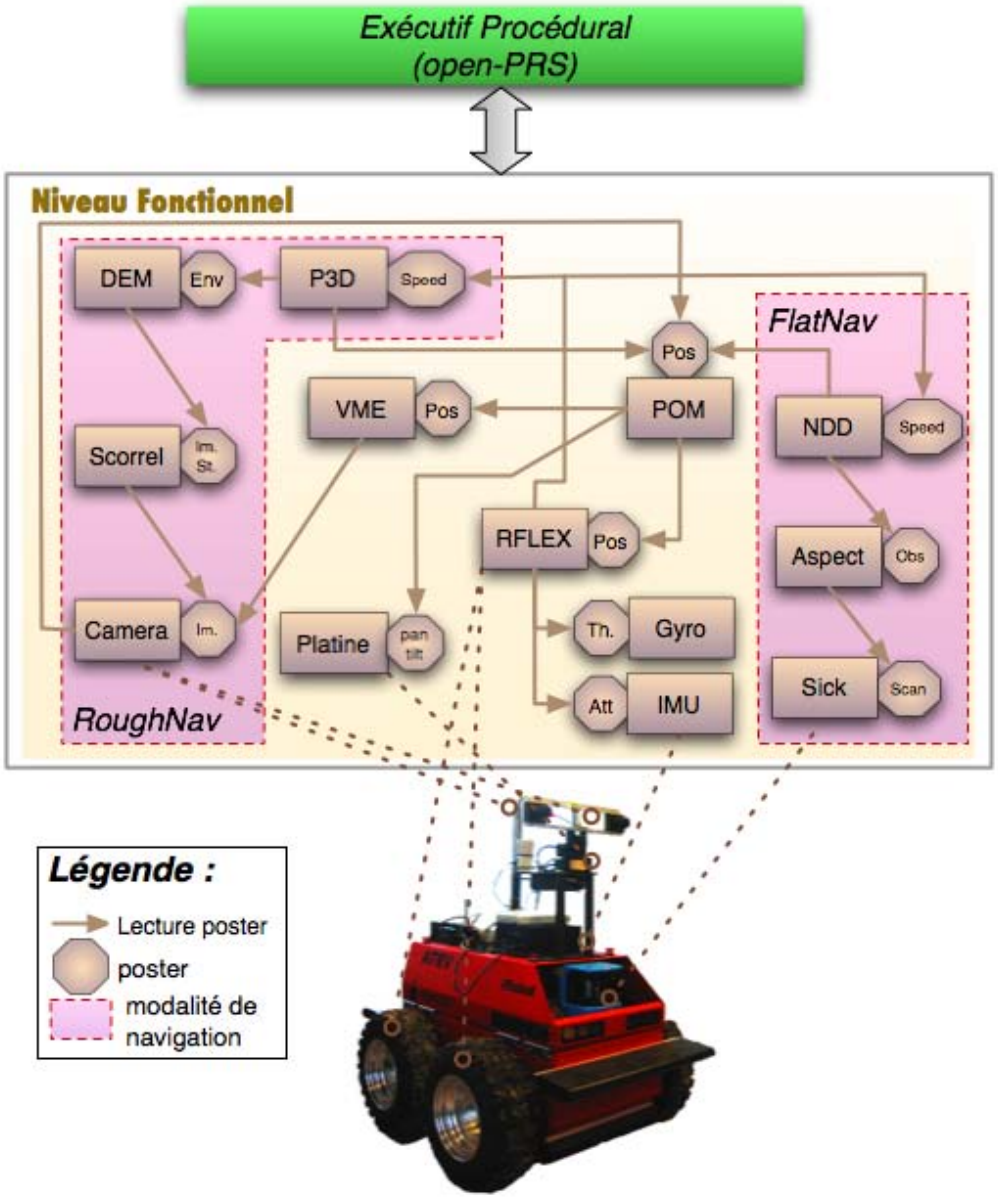


FIG. C.2 – Modules fonctionnels de Dala exploités pour les modes *FlatNav* et *RoughNav*. Les modules centraux, en dehors des zones délimitées correspondant aux deux modes de navigation, concernent la localisation. Ils restent activés quelque soit le mode de navigation exploité.

- `Gyro` lit les mesures du gyromètre d'axe z et produit une estimation de l'angle de cap θ, après compensation du biais sur la vitesse angulaire mesurée.
- `RFLEX` est le module de locomotion. Il est chargé de réaliser l'asservissement en vitesses (V_{ref}, ω_{ref}), consignes fournies ici par le module `NDD` lorsque le mode de navigation actif est *FlatNav* et par `P3D` quand c'est *RoughNav*. Ce module calcule également l'odométrie 3D, c'est pourquoi il consulte les posters des modules `Gyro` (pour l'angle de cap θ) et `IMU` (pour les estimées d'angles de roulis et tangage).
- `Platine` permet d'orienter la platine sur laquelle sont montées les caméras en tangage et en cap. Il produit un poster contenant cette orientation, destinée au module de position `POM`.
- `VME` (pour Visual Motion Estimation) effectue une estimation des déplacements relatifs du robot par *odométrie optique* (voir section A.3).
- `POM` (POsition Manager) est le module central de localisation. Il produit une estimée de la position 3D courante du robot, ainsi que des différents repères liés à ce dernier (repères des caméras par exemple), à partir des estimations complètes ou partielles réalisées par les modules producteurs de position (soient ici : `RFLEX`, `Platine` et `VME`).

C.3.2 Les modules du mode *RoughNav*

- Le module `Camera` est chargé de l'acquisition et de la manipulation des images des deux caméras, ainsi que de certaines opérations de pré-traitement pour les algorithmes de stéréo-vision (telles que la réduction, la rectification et la correction de la distorsion). A chaque paire d'images acquise est associée une position complète, grâce à une consultation de la position fournie par le module `POM`.
- `Scorrel` (pour Stereo correlation) applique les algorithmes de stéréo-vision sur la paire d'images rectifiées fournie par le module `Camera` afin de produire une image de points 3D.
- Le module `DEM` (Digital Elevation Map, équivalent de MNT en anglais) produit et met à jour un modèle numérique de terrain (ou carte d'élévations) à partir des images de points 3D fournies par `Scorrel`.
- `P3D` (pour planificateur 3D, voir section 3.2.2 page 51) est le planificateur local qui choisit une trajectoire en fonction des coordonnées du but et du placement de la structure du robot sur le MNT construit par `DEM`. L'arc de cercle choisi est traduit en consignes de vitesses (V_{ref}, ω_{ref}) qui seront régulièrement consultées par le module de locomotion `RFLEX`.

C.3.3 Les modules du mode *FlatNav*

- Le module `Sick` (du nom de la famille des capteurs laser embarqués sur les robots du LAAS-CNRS) effectue la lecture et le référencement réguliers des données du capteur laser 2D.
- Le module `Aspect` réalise la mise en forme de ces données afin qu'elles soient exploitables par le module `NDD`.
- `NDD` (de Nearness Diagram, voir section 3.2.1 page 47) est chargé de la stratégie de navigation spécifique au mode *FlatNav*. Il choisit une direction à suivre et ajuste la vitesse demandée au robot en fonction de l'encombrement de l'environnement immédiat du robot. Il produit ainsi périodiquement des consignes de vitesses (V_{ref}, ω_{ref}) qui seront régulièrement consultées par le module de locomotion `RFLEX`.

C.3.4 Les modules liés au monitoring et à la sélection de modes

La figure C.3 se focalise sur les modules mis en oeuvre pour l'estimation du mode de déplacement à appliquer et les communications nécessitées par cette fonction. Les moniteurs sont en pratique des tâches d'exécutions spécifiques intégrées au module contenant les informations à analyser. Ainsi :

- le monitoring *SurvAtt* est réalisé par une tâche d'exécution spécifique du module `P3D`, qui effectue les placements du robot sur le MNT dans le mode *RoughNav*,
- le moniteur *DAT* est mis en oeuvre dans le module de la centrale inertielle, `IMU`,
- le monitoring *LEM* est intégré au module de locomotion `RFLEX`,
- le moniteur *Locomp* est quant à lui une tâche dédiée dans le module central de localisation, `POM`.

D'autre part, c'est le bien nommé module `difmap` qui se charge de construire les *difmap* ("cartes de difficulté", voir 4.2 page 76) et de fournir ainsi les données de contexte.

Enfin, l'estimation probabiliste du mode à appliquer est réalisée dans le module `ModeEst`, client de tous les modules précédents.

C.4 openPRS et l'exécutif

`openPRS` [Ingrand, 2005] est un logiciel développé au LAAS-CNRS dédié à la supervision et le contrôle "haut-niveau" de robots mobiles autonomes, et qui exploite un langage consacré à cet usage, le langage *PRS* [Ingrand et al., 1996]. Il dispose de moyens de communication avec `GenoM`, et en particulier d'une bibliothèque d'interface d'accès aux services d'un module (envoi de requêtes à un module, réception des répliques, lecture d'un poster). Une OP (Operation Procedure) d'`openPRS` permet ainsi d'envoyer les requêtes nécessaires à tous les modules impliqués pour la réalisation d'un mode de navigation.
La figure C.4 montre l'OP appelée lorsque l'on veut activer le mode de navigation *FlatNav*.

La figure C.5 montre quant à elle l'OP appelée lorsque l'on veut activer le mode de navigation *RoughNav*.

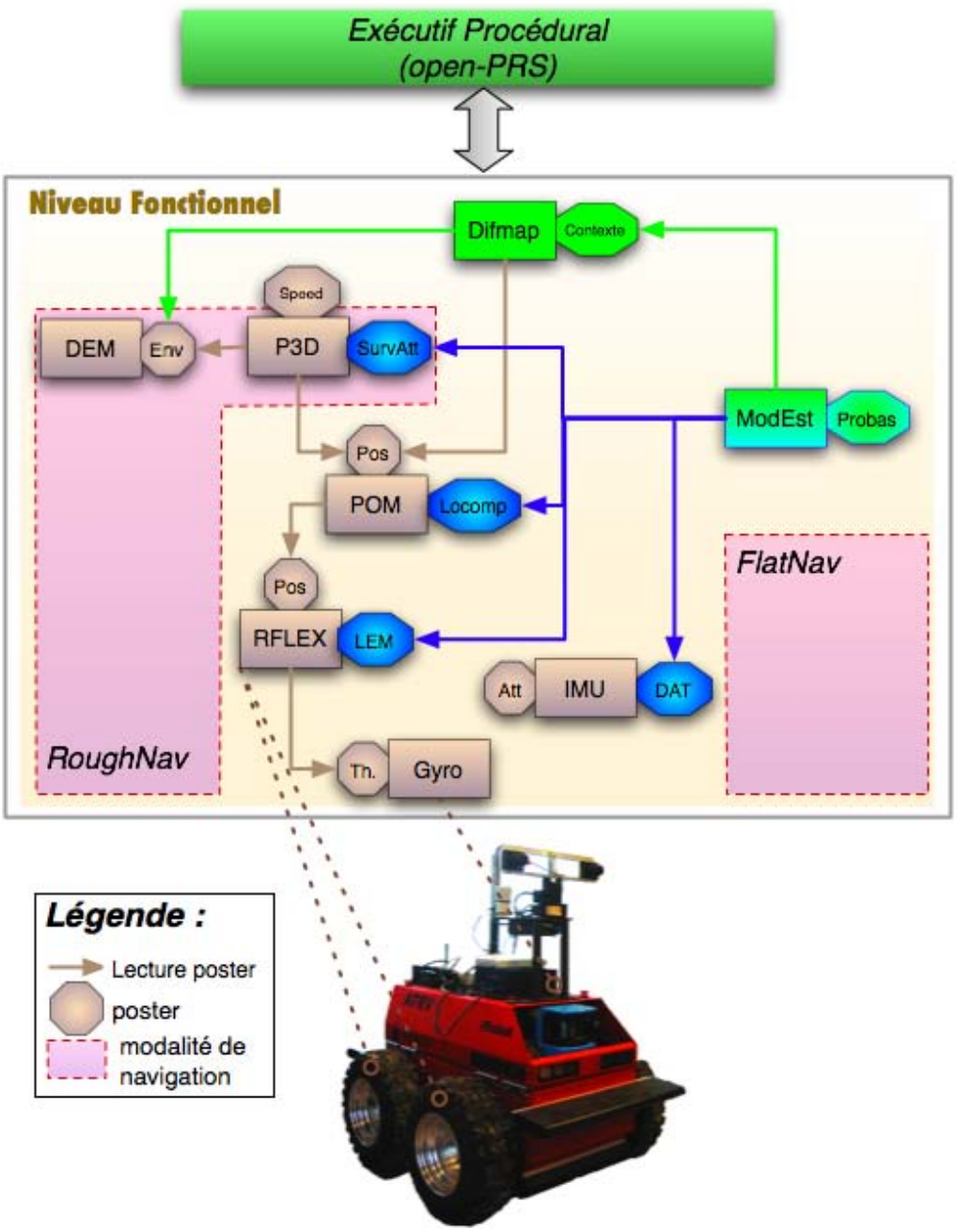


FIG. C.3 – Modules fonctionnels de Dala pour la sélection de modes et communication entre eux. Les posters en bleu contiennent les informations de *comportement* générées par une tâche d'exécution dédiée au monitoring dans chacun des modules concernés. Le module `difmap` (en vert) est quant à lui chargé de générer les données de contexte, en les rendant accessibles dans un poster. Le module `ModEst` consulte régulièrement les données de comportement fournies par les moniteurs ainsi que ces données de contexte. A partir de cela il calcule les probabilités d'application de chaque mode disponible et rend ces dernières accessibles au niveau exécutif dans un poster.

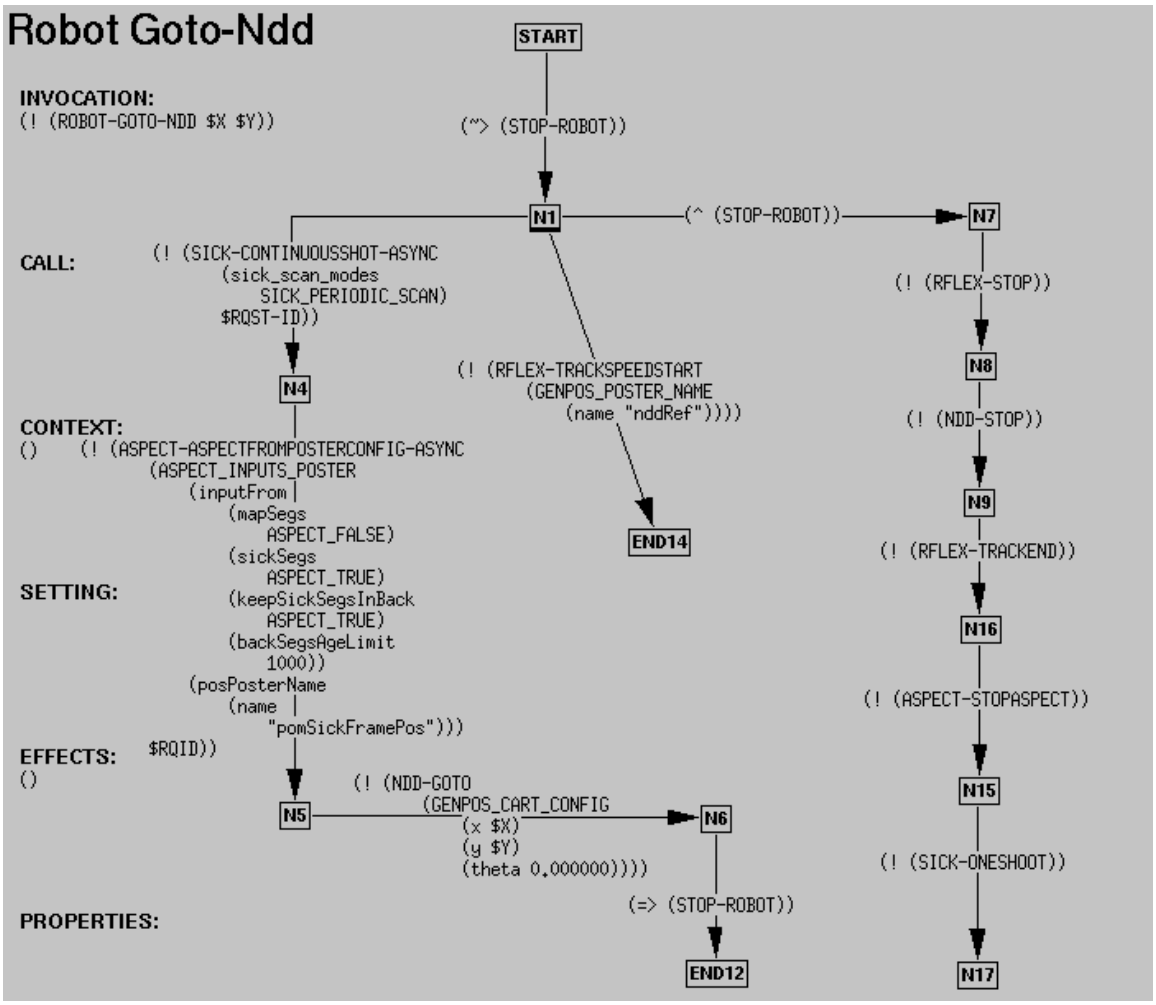


FIG. C.4 – L'OP `openPRS` pour une demande de déplacement du robot en mode *FlatNav*, afin d'atteindre le but de coordonnées ($X,$Y).

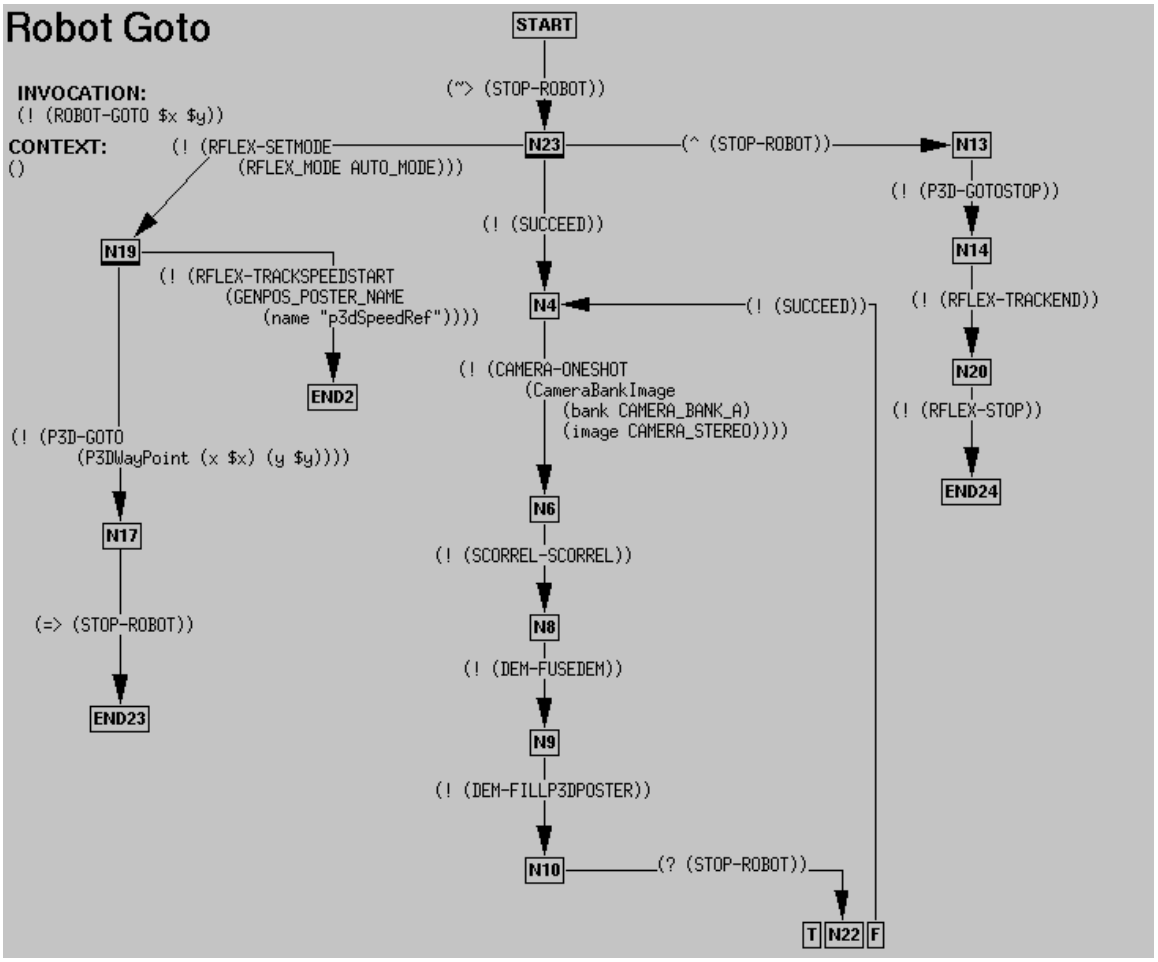


FIG. C.5 – L'OP `openPRS` pour une demande de déplacement du robot en mode *RoughNav*, afin d'atteindre le but de coordonnées ($x,$y).

Glossaire

Nous proposons ici une synthèse des définitions d'acronymes et de termes particuliers qui interviennent dans ce mémoire. En effet, ils ne sont généralement définis que lors de leur première utilisation dans le texte.

AIST : National Institute of Advanced Industrial Science and Technology, Tokyo Institute of Technology.

Amer : Terme issu du vocabulaire maritime, défini par le `Petit Larousse` comme un "objet, bâtiment fixe et visible situé sur une côte et servant de point de repère pour la navigation", d'où le mot plus "explicite" de *landmark* en anglais. Il s'agit donc plus généralement d'un élément de l'environnement qui servira de référence relative pour naviguer et se localiser.

ATRV : All Terrain Robot Vehicle : robot tout terrain de la société iRobot.

CMU : Carnegie Mellon University, Pittsburgh, PA, USA.

CNES : Centre National d'Etudes Spatiales. L'équipe de robotique mentionnée dans ce mémoire est basée à Toulouse.

DAT : *Détection d'Accident de Terrain*, moniteur contrôlant que le terrain sous le robot est plat.

EDEN : *Expérimentations de Déplacements en Environnements Naturels* : projet interne LAAS.

FDIR : Fault Detection, Identification and Recovery : Détection de Fautes, Identification et "Récupération" (ou réparation).

GPS : Global Positioning System : système de positionnement sur le globe terrestre par triangulation des signaux émis par des satellites géo-stationnaires.

HMM : Hidden Markov Model (soit modèle de Markov caché).

HUT : Helsinki University of Technology, Finlande.

IMU : Inertial Measurement Unit : centrale inertielle 3 axes, composée de 3 accéléromètres et 3 gyromètres.

JPL : Jet Propulsion Laboratory, centre de la NASA situé à Pasadena, Californie.

LAMA : A l'origine, acronyme pour Lavochkin Alcatel Model Autonomous. C'est un robot de châssis de type *Marsokhod*.

LASMEA : LAboratoire des Sciences et Matériaux pour l'Electronique, et d'Automatique, à Clermont-Ferrand.

LEM : Locomotion Efficiency Monitoring : moniteur évaluant l'efficacité de la locomotion en mode *Roulement*.
LRP : Laboratoire de Robotique de Paris, basé à Fontenay aux Roses.

MDP : Markov Decision Process (soit Processus Décisionnel de Markov).
MER : Mars Exploration Rover : programme de la NASA pour l'exploration de Mars impliquant entre autres les rovers Spirit et Opportunity. Le programme a véritablement démarré avec le lancement des sondes contenant les deux rovers jumeaux les 10 juin et 7 juillet 2003, puis les atterrissages respectifs de ces derniers sur le sol martien les 4 et 25 janvier 2004. Ils poursuivent leur mission d'exploration avec succès depuis lors.
MIT : Massachusetts Institute of Technology, à Cambridge, MA, USA.
MNT : Modèle Numérique de Terrain : carte d'élévation sur une grille cartésienne pour représenter le relief d'un terrain (DEM : Digital Elevation Map en anglais).
Monitoring : Surveillance, dans notre cas, du comportement d'un mode.

NASA : National Aeronautics and Space Administration : Agence nationale aéronautique et espace des USA.

Péristaltisme : Mode de locomotion dans lequel un robot équipé de mobilités internes "rampe" à la manière d'une chenille, faisant avancer successivement chacun de ses essieux, un par un.
PHCA : Probabilistic Hierarchical Constraint-based Automata, introduit par Brian C. Williams du MIT [Mikaelian et al., 2005].
POMDP : Partially Observable Markov Decision Process (Processus Décisionnel de Markov Partiellement Observable).

R2M : *Rover Multi-Modes pour une haute mobilité sur terrain accidenté* [Martinet et al., 2006, Martinet et al., 2005], projet Robea (novembre 2003 à début avril 2006) impliquant le LRP, le LASMEA, le LAAS et le centre DGA d'Angers (ETAS). Il a vu la conception et la naissance du mini-rover Hylos 2.
Robea : Programme National ROBotique et Entités Artificielles.
Rover : Terme parfois utilisé pour désigner simplement un robot mobile à roues évoluant en environnement naturel. On parle souvent ainsi de rovers planétaires pour les robots mobiles chargés de missions d'exploration de planètes telles que Mars : Rocky ou les deux robots de MER Spirit et Opportunity en sont des exemples bien connus.

Skid-steering rover : Robot à roues de type "char" (i.e. sans roues directionnelles).
SLAM : Simultaneous Localization and Mapping : méthode consistant à effectuer simultanément la localisation et la construction de carte pour un robot, l'une se servant des données de l'autre.
SurvAtt : Moniteur de Surveillance de l'attitude. Il repose sur la comparaison en ligne de l'*observation* de l'attitude (et de la configuration) du robot avec la *prédiction* issue du placement du robot sur le MNT.

TALC : *Terrain Adaptive Locomotion Control*, pourrait également être appelée "terrain-based locomotion control" : commande en vitesse pour un rover de type *skid-sterring* tenant compte du relief du terrain (voir [Peynot et al., 2003] et section 3.3.1).

UAV : Unmanned Aerial Vehicle : Engin aérien autonome tel qu'un drone.

Bibliographie

[Alami et al., 1998] R. Alami, R. Chatila, S. Fleury, M. Ghallab, et F. Ingrand (1998). An architecture for autonomy. *International Journal of Robotic Research, Special Issue on Integrated Architectures for Robot Control and Programming.*

[Andrade et al., 1998] G. Andrade, F. B. Amar, P. Bidaud, et R. Chatila (1998). Modeling robot-soil interaction for planetary rover motion control. Dans *IEEE/RSJ International Conference on Robots and Intelligent Systems.*

[Arnaud et al., 2005] E. Arnaud, E. Mémin, et B. Cernuschi-Frias (2005). Conditional filters for image sequence based tracking - application to point tracking. *IEEE Transactions on Image Processing*, 14(1) :63–79.

[Aviña, 2005] J. G. Aviña (2005). *Navigation visuelle d'un robot mobile dans un environnement d'extérieur semi-structuré.* Thèse de doctorat, Institut National Polytechnique de Toulouse.

[Bernstein et al., 2001] D. Bernstein, S. Zilberstein, R. Washington, et J. Bresina (2001). Planetary rover control as a Markov Decision Process. Dans *International Symposium on Artificial Intelligence, Robotics and Automation in Space, St Hubert, Qc (Canada).*

[Biesiadecki et al., 2005] J. J. Biesiadecki, C. Leger, et M. W. Maimone (2005). Tradeoffs between directed and autonomous driving on the mars exploration rovers. Dans *12th International Symposium of Robotics Research.*

[Bloch, 1996] I. Bloch (1996). Incertitude, imprécision et additivité en fusion de données : Point de vue historique. *Traitement du Signal*, 13(4) :267–288.

[Boizard et al., 2005] J. Boizard, A. Naoulou, J. Fourniols, M. Devy, T. Sentenac, et P. Lacroix (2005). FPGA based architectures for real time computation of the census transform and correlation in various stereovision contexts. Dans *7th International Workshop on Electronics, Control, Modelling, Measurement and Signals (ECMS'2005).*

[Bonnafous et al., 2001] D. Bonnafous, S. Lacroix, et T. Siméon (2001). Motion generation for a rover on rough terrains. Dans *IEEE/RSJ International Conference on Robots and Intelligent Systems.*

[Borenstein et al., 1996a] J. Borenstein et L. Feng (1996a). Gyrodometry : A new method for combining data from gyros and odometry in mobile robots. Dans *IEEE International Conference on Robotics and Automation.*

[Borenstein et al., 1996b] J. Borenstein et L. Feng (1996b). Measurement and correction of systematic odometry errors in mobile robot. *IEEE Transactions on Robotics and Automation.*

[Borenstein et al., 1991] J. Borenstein et Y. Koren (1991). The Vector Field Histogram - Fast obstacle avoidance for mobile robots. *IEEE Journal of Robotics and Automation.*

[Bosch et al., 2006] S. Bosch, S. Lacroix, et F. Caballero (2006). Autonomous detection of safe landing areas for an uav from monocular images. Dans *Submitted to the IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Brèthes, 2005] L. Brèthes (2005). *Suivi visuel par filtrage particulaire. Application à l'interaction homme–robot.* Thèse de doctorat, Université Paul Sabatier, Toulouse.

[Castelnovi et al., 2005] M. Castelnovi, R. Arkin, et T. Collins (2005). Reactive speed control system based on terrain roughness detection. Dans *IEEE International Conference on Robotics and Automation.*

[Chatila et al., 1995] R. Chatila et S. Lacroix (1995). Adaptive navigation for autonomous mobile robots. Dans *International Symposium on Robotics Research.*

[CMU-NASA, 2006] CMU-NASA (2003-2006). *Life in the Atacama.* http ://www.frc.ri.cmu.edu/atacama/.

[Coronado-Vergara et al., 2005] J. Coronado-Vergara, G. A. Cervantes, M. Devy, et C. Parra (2005). Towards landmine detection using artificial vision. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[da Silveira et al., 2005] da M. Silveira et L. Trave-Massuyes (2005). Diagnosis for autonomous satellites : State of the art. Rapport technique, LAAS-CNRS.

[Dalgalarrondo et al., 2004] A. Dalgalarrondo, D. Dufourd, et D. Filliat (2004). Controlling the autonomy of a reconnaissance robot. Dans *Symposium SPIE Defense & Security, session Unmanned Ground Vehicle Technology VI.*

[Dearden et al., 2002] R. Dearden et D. Clancy (2002). Particle filters for real-time fault detection in planetary rovers. Dans *Proceedings of the 12th International Workshop on Principles of Diagnosis.*

[Delpech et al., 1998] M. Delpech, L. Rastel, et M. Lamboley (1998). Enhanced path planning and localization techniques for autonomous planetary rovers. Dans *5th ESA Workshop on Advanced Space Technologies for Robotics and Automation.*

[Dubuisson, 2001] B. Dubuisson (2001). *Automatique et statistiques pour le diagnostic.* Hermes Science.

[Endo et al., 1999] G. Endo et S. Hirose (1999). Study on Roller-Walker (system integration and basic experiments). Dans *Proceedings of the 1999 IEEE International Conference on Robotics and Automation.*

[Endo et al., 2000] G. Endo et S. Hirose (2000). Study on Roller-Walker (multi-mode steering control and self-contained locomotion). Dans *Proceedings of the 2000 IEEE International Conference on Robotics and Automation.*

[Espiau et al., 1992] B. Espiau, F. Chaumette, et P. Rives (1992). A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation.*

[Estier et al., 2000] T. Estier, Y. Crausaz, B. Merminod, M. Lauria, R. Piguet, et R. Siegwart (2000). An innovative space rover with extended climbing abilities. Dans *Proceedings of Space and Robotics, the Fourth International Conference and Exposition on Robotics in Challenging Environments.*

[Fernández, 2000] J. L. Fernández (2000). *Supervision, detection, diagnosis and exception recovery in autonomous mobile robots.* Thèse de doctorat, Universidad de Vigo, Departamiento de Ingeniería de Sistemas y Automática.

[Fleury et al., 1997] S. Fleury, M. Herrb, et R. Chatila (1997). GenoM : a tool for the specification and the implementation of operating modules in a distributed robot architecture. Dans *Proceedings of the International Conference on Intelligent Robots and Systems.*

[Fleury et al., 2005] S. Fleury, M. Herrb, et A. Mallet (2005). *GenoM : Generator of Modules for Robots.* http ://softs.laas.fr/openrobots/tools/genom.php.

[Funiak et al., 2003] S. Funiak et B. Williams (2003). Multi-modal particle filtering for hybrid systems with autonomous mode transitions. Dans *Proceedings of the 13th International Workshop on Principles of Diagnosis.*

[Gancet et al., 2003] J. Gancet et S. Lacroix (2003). PG2P : a perception-guided path planning approach for long range autonomous navigation in unknown natural environments. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Golberg et al., 2002] S. Golberg, M. Maimone, et L. Matthies (2002). Stereo vision and rover navigation software for planetary exploration. Dans *IEEE Aerospace Conference.*

[Grand, 2004] C. Grand (2004). *Optimisation et commande des modes de déplacement des systèmes locomoteurs hybrides roue-patte. Application au robot Hylos.* Thèse de doctorat, Université Paris 6.

[Grand et al., 2002] C. Grand, F. BenAmar, et P. Bidaud (2002). Kinematic analysis and stability optimisation of a reconfigurable legged-wheeled mini-rover. Dans *SPIE'02 : Unmanned ground-vehicle technology IV.*

[Grand et al., 2004] C. Grand, F. BenAmar, F. Plumet, et P. Bidaud (2004). Decoupled control of posture and trajectory of the hybrid wheel-legged robot Hylos. Dans *IEEE International Conference on Robotics and Automation.*

[Iagnemma et al., 2000a] K. Iagnemma et S. Dubowsky (2000a). Mobile Robot Rough-Terrain Control (RTC) for planetary exploration. Dans *Proceedings of ASME IDETC/CIE : 26th Biennial Mechanisms and Robotics Conference.*

[Iagnemma et al., 2000b] K. Iagnemma et S. Dubowsky (2000b). Vehicle wheel-ground contact angle estimation : with application to mobile robot traction control. Dans *7th International Symposium on Advances in Robot Kinematics.*

[Ingrand, 2005] F. Ingrand (2005). *OpenPRS : an open source version of PRS (Procedural Reasoning Systems).* http ://softs.laas.fr/openrobots/tools/openprs.php.

[Ingrand et al., 1996] F. Ingrand, R. Chatila, R. Alami, et F. Robert (1996). PRS : A high level supervision and control language for autonomous mobile robots. Dans *IEEE International Conference on Robotics and Automation.*

[JPL-NASA, 2006] JPL-NASA (2003-2006). *Mars Exploration Rover Mission.* http ://marsrovers.nasa.gov.

[Kamimura et al., 2002] A. Kamimura, E. Yoshida, S. Murata, H. Kurokawa, K. Tomita, et S. Kokaji (2002). A self-reconfigurable modular robot (MTRAN) - hardware and motion generation software. Dans *International Symposium on Distributed Autonomous Robotic Systems.*

[Kelly et al., 1997] A. Kelly et A. Stentz (1997). Analysis of requirements for high speed rough terrain autonomous mobility. Part I : Throughput and response. Dans *Proceedings of the 1997 IEEE International Conference on Robotics and Automation.*

[Kurokawa et al., 2002] H. Kurokawa, A. Kamimura, E. Yoshida, K. Tomita, S. Murata, et S. Kokaji (2002). Self-reconfigurable modular robot (M-TRAN) and its motion design. Dans *Seventh International Conference on Control, Automation, Robotics And Vision (ICARCV'02).*

[Lacroix et al., 1995] S. Lacroix et R. Chatila (1995). Motion and perception strategies for outdoor mobile robot navigation in unknown environments. Dans O. Khatib et J. Salisbury, editors, *Experimental Robotics IV*, volume 223 of *Lecture Notes in Computer and Information Science*, pages 538–547. Springer.

[Lacroix et al., 2004] S. Lacroix et I.-K. Jung (2004). Simultaneous localization and mapping with stereovision. Dans *5th symposium on Intelligent Autonomous Vehicles.*

[Lacroix et al., 2002] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, et R. Chatila (2002). Autonomous rover navigation on unknown terrains : Functions and integration. *International Journal of Robotics Research.*

[Laumond, 2001] J. Laumond (2001). *La robotique mobile.* Hermes.

[Lemonde, 2005] V. Lemonde (2005). *Stéréovision Embarquée sur Véhicule : de l'Auto-Calibrage à la Détection d'Obstacles.* Thèse de doctorat, Institut National des Sciences Appliquées de Toulouse.

[Mallet, 2001] A. Mallet (2001). *Localisation d'un robot mobile autonome en environnements naturels.* Thèse de doctorat, Institut National Polytechnique de Toulouse - LAAS-CNRS.

[Mallet et al., 2000] A. Mallet, S. Lacroix, et L. Gallo (2000). Position estimation in outdoor environments using pixel tracking and stereovision. Dans *IEEE International Conference on Robotics and Automation.*

[Martinet et al., 2006] P. Martinet, B. Thuilot, F. Hao, M. Berducat, C. Cariou, C. Debain, R. Lenain, C. Tessier, A. Godin, F. B. Amar, F. Plumet, C. Grand, G. Besseron, S. Lacroix, et T. Peynot (2006). R2M : Rover Multi-Modes pour une haute mobilité sur terrain accidenté. Dans *Journées Bilan ROBEA.*

[Martinet et al., 2005] P. Martinet, B. Thuilot, F. Hao, M. Berducat, C. Cariou, C. Debain, C. Tessier, A. Godin, C. Bouzgarrou, J. C. Fauroux, F. B. Amar, F. Plumet, C. Grand, G. Besseron, T. Peynot, et S. Lacroix (2005). R2M : Rover Multi-Modes pour une haute mobilité sur terrain accidenté. Dans *3èmes Journées du Programme ROBEA (ROBEA'2005).*

[Mateus et al., 2005] D. Mateus, G. Avina, et M. Devy (2005). Robot visual navigation in semi-structured outdoor environments. Dans *IEEE International Conference on Robotics and Automation.*

[Matthies et al., 2003] L. Matthies et A. Rankin (2003). Negative obstacle detection by thermal signature. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Metni et al., 2005] N. Metni, J.-M. Pflimlin, T. Hamel, et P. Souères (2005). Attitude and gyro bias estimation for a flying UAV. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Michaud et al., 2003] F. Michaud, D. Létourneau, J.-F. Paré, M.-A. Legault, R. Cadrin, M. Arsenault, Y. Bergeron, M.-C. Tremblay, F. Gagnon, M. Millette, P. Lepage, Y. Morin, et S. Caron (2003). Azimut - A leg-track-wheel robot. Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Mikaelian et al., 2005] T. Mikaelian, B. Williams, et M. Sachenbacher (2005). Autonomous diagnosis based on software-extended behavior models. Dans *Proceedings of the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space - iSAIRAS.*

[Minguez et al., 2001] J. Minguez, L. Montano, N. Simeon, et R. Alami (2001). Global Nearness Diagram Navigation (GND). Dans *IEEE International Conference on Robotics and Automation.*

[Minguez et al., 2004] J. Minguez, J. Osuna, et L. Montano (2004). A "divide and conquer" strategy based on situations to achieve reactive collision avoidance in troublesome scenarios. Dans *IEEE International Conference on Robotics and Automation.*

[Morisset, 2002] B. Morisset (2002). *Vers un robot au comportement robuste. Apprendre à combiner des modalités sensori-motrices complémentaires.* Thèse de doctorat, Université Paul Sabatier de Toulouse.

[Payton et al., 1992] D. W. Payton, D. Keirsey, D. M. Kimble, J. Krozel, et J. K. Rosenblatt (1992). Do whatever works : A robust approach to fault-tolerant autonomous control. *Journal of Applied Intelligence*, 2(3) :225–250.

[Peynot, 2002] T. Peynot (2002). Contrôle de locomotion d'un robot tout-terrain. Master's thesis, Institut National Polytechnique de Toulouse, ENSEEIHT.

[Peynot et al., 2003] T. Peynot et S. Lacroix (2003). Enhanced locomotion control for a planetary rover. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Peynot et al., 2005] T. Peynot et S. Lacroix (2005). A probabilistic framework to monitor a multi-mode outdoor robot. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[Pirjanian et al., 2002] P. Pirjanian, C. Leger, E. Mumm, B. Kennedy, M. Garrett, H. Aghazarian, S. Farritor, et P. Schenker (2002). Distributed control for a modular, reconfigurable cliff robot. Dans *Proceedings of the 2002 IEEE International Conference on Robotics and Automation.*

[Postaire, 1987] J.-G. Postaire (1987). *De l'image à la décision.* DUNOD informatique.

[Py, 2005] F. Py (2005). *Contrôle d'Exécution dans une Architecture Hiérarchisée pour Systèmes Autonomes.* Thèse de doctorat, Université Paul Sabatier, Toulouse.

[Rives et al., 2001a] P. Rives et M. Devy (2001a). *La robotique mobile*, chapitre Perception pour la navigation et la commande, pages 199–257. Hermes.

[Rives et al., 2001b] P. Rives et M. Devy (2001b). *La robotique mobile*, chapitre Perception pour la localisation, pages 141–198. Hermes.

[Russell et al., 2003a] S. Russell et P. Norvig (1995, 2003a). *Artificial Intelligence. A Modern Approach*, chapitre 15 : Probabilistic Reasoning over Time. Prentice Hall Series in Artificial Intelligence, second edition.

[Russell et al., 2003b] S. Russell et P. Norvig (1995, 2003b). *Artificial Intelligence. A Modern Approach.* Prentice Hall Series in Artificial Intelligence, second edition.

[Russell et al., 2003c] S. Russell et P. Norvig (1995, 2003c). *Artificial Intelligence. A Modern Approach*, chapitre 21 : Reinforcement Learning. Prentice Hall Series in Artificial Intelligence, second edition.

[Russell et al., 2003d] S. Russell et P. Norvig (1995, 2003d). *Artificial Intelligence. A Modern Approach*, chapitre 14 : Probabilistic Reasoning. Prentice Hall Series in Artificial Intelligence, second edition.

[Seraji et al., 2001a] H. Seraji, A. Howard, et E. Tunstel (2001a). Safe navigation on hazardous terrain. Dans *IEEE International Conference on Robotics and Automation.*

[Seraji et al., 2001b] H. Seraji, A. Howard, et E. Tunstel (2001b). Terrain-based navigation of planetary rovers : A fuzzy logic approach. Dans 6^th *International Symposium on Artificial Intelligence, Robotics and Automation in Space : i-SAIRAS.*

[Singh et al., 2000] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verna, A. Yahja, et K. Schwehr (2000). Recent progress in local and global traversability for planetary rovers. Dans *IEEE International Conference on Robotics and Automation.*

[Sukkarieh, 2000] S. Sukkarieh (2000). *Low Cost, High Integrity, Aided Inertial Navigation Systems for Autonomous Land Vehicles*. Thèse de doctorat, Australian Centre for Field Robotics, Department of Mechanical and Mechatronic Engineering, University of Sydney.

[Thrun et al., 2005] S. Thrun, W. Burgard, et D. Fox (2005). *Probabilistic Robotics*. The MIT Press.

[Verma et al., 2001] V. Verma, J. Langford, et R. Simmons (2001). Non-parametric fault identification for space rovers. Dans *Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space - iSAIRAS*.

[Verma et al., 2002] V. Verma, R. Simmons, et J. Fernandez (2002). Probabilistic models for monitoring and fault diagnosis. Dans *2nd IARP/IEEE-RAS Joint Workshop on Technical Challenge for Dependable Robots in Human Environments*.

[Wettergreen et al., 2005] D. Wettergreen, N. Cabrol, V. Baskaran, F. Calderon, S. Hueys, D. Jonal, A. Luders, D. Pane, T. Smith, J. Teza, P. Tompkins, D. Villa, C. Williams, et M. Wagner (2005). Second experiments in the robotic investigation of life in the Atacama desert of Chile. Dans *Proceedings of the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space - iSAIRAS*.

[White, 1993] D. White (1993). *Markov Decision Processes*. John WILEY & Sons.

Résumé

Le déplacement entièrement autonome d'un robot mobile en environnements naturels est un problème encore loin d'être résolu. Il nécessite la mise en oeuvre de fonctionnalités permettant de réaliser le cycle perception/décision/action, que nous distinguons en deux catégories : *navigation* (perception et décision sur le mouvement à réaliser) et *locomotion* (réalisation du mouvement). Pour pouvoir faire face à la grande diversité de situations que le robot peut rencontrer en environnement naturel, il peut être primordial de disposer de plusieurs types de fonctionnalités complémentaires, constituant autant de *modes de déplacement* possibles. En effet, de nombreuses réalisations de ces derniers ont été proposées dans la littérature ces dernières années mais aucun ne peut prétendre permettre d'exécuter un déplacement autonome en toute situation.
Par conséquent, il semble judicieux de doter un robot mobile d'extérieur de plusieurs modes de déplacement complémentaires. Dès lors, ce dernier doit également disposer de moyens de choisir en ligne le mode le plus approprié.

Dans ce cadre, cette thèse propose une mise en oeuvre d'un tel système de sélection de mode de déplacement, réalisée à partir de deux types de données : une observation du *contexte* pour déterminer dans quel type de situation le robot doit réaliser son déplacement et une surveillance du *comportement* du mode courant, effectuée par des *moniteurs*, et qui influence les transitions vers d'autres modes lorsque le comportement du mode actuel est jugé non satisfaisant.

Ce manuscrit présente donc : un formalisme probabiliste d'estimation du mode à appliquer, des modes de navigation et de locomotion exploités pour réaliser le déplacement autonome, une méthode de représentation qualitative du terrain (reposant sur l'évaluation d'une *difficulté* calculée après placement de la structure du robot sur un modèle numérique de terrain), et des moniteurs surveillant le comportement des modes de déplacement utilisés (évaluation de l'efficacité de la locomotion par roulement, surveillance de l'attitude et de la configuration du robot...).

Quelques résultats expérimentaux de ces éléments intégrés à bord de deux robots d'extérieur différents sont enfin présentés et discutés.

Mots-clés: Robotique mobile en environnement naturel, Modes de Navigation, Modes de Locomotion, Robotique probabiliste, Modèle de Markov caché, Monitoring

Abstract

Autonomous navigation and locomotion of a mobile robot in natural environments remain a rather open issue. Several functionalities are required to complete the usual perception/decision/action cycle. They can be divided in two main categories : *navigation* (perception and decision about the movement) and *locomotion* (movement execution). In order to be able to face the large range of possible situations in natural environments, it is essential to make use of various kinds of complementary functionalities, defining various *navigation and locomotion modes*. Indeed, a number of navigation and locomotion approaches have been proposed in the litterature for the last years, but none can pretend being able to achieve autonomous navigation and locomotion in every situation.
Thus, it seems relevant to endow an outdoor mobile robot with several complementary navigation and locomotion modes. Accordingly, the robot must also have means to select the most appropriate mode to apply.

This thesis proposes the development of such a navigation/locomotion mode selection system, based on two types of data : an observation of the *context* to determine in what kind of situation the robot has to achieve its movement and an evaluation of the *behavior* of the current mode, made by *monitors* which influence the transitions towards other modes when the behavior of the current one is considered as non satisfying.

Hence, this document introduces a probabilistic framework for the estimation of the mode to be applied, some navigation and locomotion modes used, a qualitative terrain representation method (based on the evaluation of a *difficulty* computed from the placement of the robot's structure on a digital elevation map), and monitors that check the behavior of the modes used (evaluation of rolling locomotion efficiency, robot's attitude and configuration watching...).

Some experimental results obtained with those elements integrated on board two different outdoor robots are presented and discussed.

Keywords: Outdoor Mobile Robotics, Navigation Modes, Locomotion Modes, Probabilistic Robotics, Hidden Markov Model, Monitoring