Navigation de Robots Mobiles par Mémoire Sensorielle

Jonathan Courbon

N° d'ordre: 000

THÈSE

présentée

DEVANT L'UNIVERSITÉ BLAISE PASCAL - CLERMONT II

pour obtenir

le grade de : *DOCTEUR DE L'UNIVERSITÉ BLAISE PASCAL*
Mention: Vision et Robotique

par

JONATHAN COURBON

Équipe d'accueil : GRAVIR (LASMEA, UMR 6602 CNRS)
École Doctorale : Sciences Pour l'Ingénieur

Titre de la thèse :

Navigation de Robots Mobiles par Mémoire Sensorielle

COMPOSITION DU JURY

M. :	El Mustapha	Mouaddib	Président
MM. :	Simon	Lacroix	Rapporteurs
	Patrick	Rives	
M. :	Philippe	Martinet	Directeur de thèse
MM. :	Nicolas	Guénard	Examinateurs
	Youcef	Mezouar	

Table des matières

Chapitre 1

Introduction

Le terme *robot* est entré depuis longtemps dans le langage courant. Il est, cependant, employé pour désigner des choses parfois très différentes. En informatique, un robot est une composante d'un moteur de recherche qui parcourt internet, afin d'alimenter en données son index. Un robot inclut plus fréquemment une structure mécanique. Il peut, par exemple, servir à mixer ou pétrir des aliments (voir Fig. 1.1 (a)) ou à nettoyer le sol tandis que d'autres robots sont dédiés au divertissement comme le grand Éléphant de Nantes (voir Fig. 1.1 (b)). Dans les laboratoires de recherche, on peut également rencontrer des robots bio-mimétiques à la forme de salamandre, de poisson ou d'insecte (voir Fig. 1.1 (c), (d), (e)). Le terme *animat* (pour "Animal artificiel") est alors employé. Dans les œuvres littéraires et cinématographiques de science fiction, les robots et les machines [1] sont parfois dotés d'une intelligence hors norme. Dans *Transformers* et dans *Matrix* (Fig. 1.1 (f), (g)), les machines sont au pouvoir. Bender, du dessin animé *Futurama*, est un robot tordeur qui boit de l'alcool, fume des cigares et qui peut être égoïste, pervers et vulgaire. C-3PO de *Star Wars* est un robot d'apparence humaine "maîtrisant plus de six millions de formes de communication". Dans le film *I, Robot*, se déroulant en 2035, les robots assistent les êtres humains avant de se rebeller. Ce comportement est bien sûr en désaccord avec les trois célèbres lois régissant l'éthique d'un robot [2] :

Loi I : Un robot ne peut porter atteinte à un être humain ni, restant passif, laisser cet être humain exposé au danger.

Loi II : Un robot doit obéir aux ordres donnés par les êtres humains, sauf si de tels ordres sont en contradiction avec la Première Loi.

Loi III : Un robot doit protéger son existence dans la mesure où cette protection n'entre pas en contradiction avec la Première ou la Deuxième Loi.

[1] On différencie alors les machines des robots : "un robot est une machine fabriquée pour imiter de son mieux l'être humain" (Isaac Asimov).

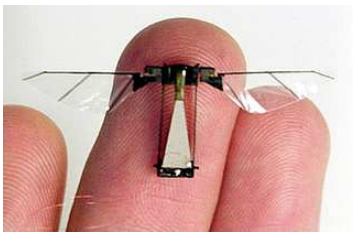
[2] Ces lois ont été décrites par l'auteur de science-fiction I. Asimov dans la nouvelle "Menteur" publiée en mai 1941.

(a) Robot ménager

(b) Éléphant de Nantes

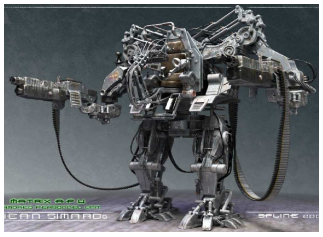
(c) Robot Salamandre (EPFL)

(d) Robot Carpe (université de l'Essex)

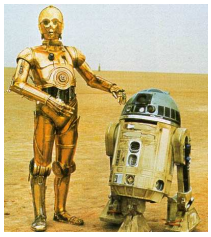


(e) Robot Mouche (université d'Harvard)

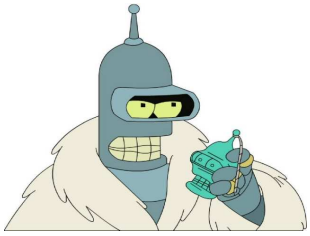
(f) Une machine de *Transformers*



(g) Une machine de *Matrix*



(h) C-3PO et R2-D2 de *Star Wars*



(i) Bender de *Futurama*

(j) Un robot de *I, Robot*

FIG. 1.1 – Des robots ?

À l'heure actuelle, nous sommes encore loin de la robotique de science-fiction car l'autonomie des robots est relativement limitée. Afin de clarifier nos propos, nous utiliserons la définition suivante pour un robot :

> *Un robot est un dispositif mécanique équipé d'actionneurs et de capteurs, capable d'effectuer des tâches de façon automatique.*

Les premiers robots ont été développés au milieu du XX$^{\text{ème}}$ siècle afin de répondre à des besoins industriels. Le rôle de ces robots dits *manipulateurs* est d'accomplir des opérations d'usinage, d'assemblage, de déplacement ... La base de ces robots est fixe par rapport à l'environnement (voir Fig. 1.2). Ils effectuent des tâches répétitives et sont donc réduits à l'état d'automates. De plus, ils agissent dans un environnement étudié spécifiquement pour chaque robot et chaque application. Contrairement aux robots manipulateurs, les robots mobiles, apparus un peu plus tard, effectuent de grands déplacements dans des environnements a priori inconnus. Les applications principales sont l'exécution de tâches dans des zones difficiles d'accès (espace, exploration planétaire), des zones dangereuses (zones contaminées) ou des zones à petites échelles (robotique chirurgicale) ainsi que l'exécution de tâches difficiles à réaliser (déplacement de charges lourdes par exemple). Jusqu'aux années 2000, les recherches ont porté principalement sur les robots à roues, robots simples du point de vue mécanique et se déplaçant dans un espace limité à deux dimensions. Pour cette raison, le terme *robotique mobile* est souvent employé pour désigner uniquement les robots à roues. De notre côté, il sera utilisé pour désigner tous les robots à base mobile dont les robots aériens, sous-marins, marcheurs ou bipèdes (voir Fig. 1.3).

Avec les progrès dans le domaine de l'informatique, l'autonomie des robots a augmenté. Les progrès informatiques couplés à la maturité des recherches sur les robots à roues ont permis d'envisager des applications de plus en plus complexes. En l'absence d'intervention humaine dans le processus de décision, ces robots sont alors qualifiés d'*autonomes*. Trois types d'autonomie sont ici impliqués : l'autonomie de puissance (le robot possède ses propres réserves énergétiques afin de se déplacer), l'autonomie sensorielle (le robot possède ses propres capacités de perception) et l'autonomie décisionnelle (le robot décide des actions qu'il doit réaliser). Une grande partie des efforts dans le domaine de la robotique ont donc porté et portent toujours sur l'amélioration de l'autonomie et des capacités d'adaptation des systèmes robotiques.

Dans la suite de ce chapitre, nous discuterons tout d'abord la navigation des robots mobiles. Nous nous intéresserons ensuite aux mécanismes impliqués dans la navigation chez les êtres humains (Section 1.2). Enfin, nous présenterons le cadre de nos travaux de thèse dans la Section 1.3.

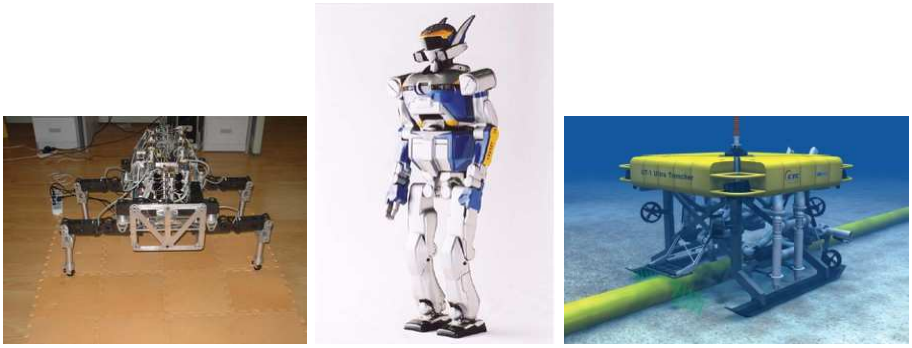
1.1 Navigation des robots mobiles

La navigation, terme emprunté au domaine de la marine, consiste à diriger de manière sûre le robot vers une destination. L'objectif de nombreux travaux de recherche

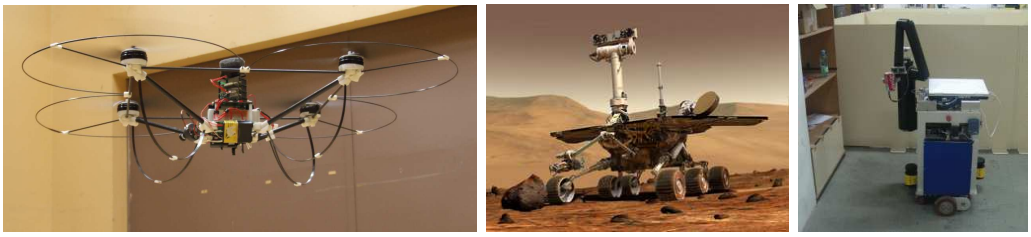
(a) RX 200 (Staübli) (b) H4 (LIRMM)

FIG. 1.2 – Des robots manipulateurs à structure (a) sérielle, (b) parallèle.

(a) RobuCab (Robosoft) (b) Pekee (Wany Robotics) (c) Pioneer 3-AT (ActivMedia) (d) Trilobit (Electrolux)



(e) MiniQuad (Huazhong University) (f) HRP2 (AIST) (g) Ut1 Ultra Trencher (Soil Machine Dynamics)



(h) X4-Flyer (CEA-List) (i) MER (NASA) (j) SAM (CEA-List)

FIG. 1.3 – Des robots mobiles.

en robotique est de rendre cette navigation la plus autonome possible. Plusieurs domaines d'applications sont concernés par cette problématique : systèmes de transport intelligent, services, assistance, surveillance. . ..

Dans le domaine des transports, des propositions de véhicules urbains en libre service voient le jour afin, entre autres, de désengorger les centres-villes mais également afin de diminuer la pollution atmosphérique. Ce type de transport, situé entre les transports en commun, les véhicules classiques et les déplacements pédestres apporte une complémentarité par rapport aux services actuels. Un parc de voitures électriques en libre service, Liselec [3], est ainsi proposé dans l'agglomération de La Rochelle depuis une dizaine d'années. De même, la mairie de Paris envisage de mettre en place un système équivalent au Vélib' mais avec des véhicules électriques (Autolib'). Une solution plus ambitieuse consiste à munir ces véhicules d'une intelligence embarquée afin de les rendre totalement autonomes. On parle alors de *cybercar* [Fraichard 05]. Le cybercar permettrait d'optimiser les déplacements, d'augmenter le confort des utilisateurs mais également d'accroître le nombre de gares. Bien entendu, une telle approche pose de nombreux problèmes du point de vue technologique. Parmi ces problèmes, le déplacement des véhicules doit être totalement sûr et autonome et la flotte des véhicules doit être gérée de manière efficace.

Une application des robots mobiles relativement différente est la téléprésence et la télésurveillance dans les *maisons communicantes*. Ces maisons sont munies d'outils adaptés afin de faciliter la réalisation des différentes tâches et de rendre les habitations plus conviviales. Entre autres, elles sont connectées à l'extérieur, équipées d'objets communicants et ont un comportement autonome. Elles peuvent être également équipées d'aspirateurs automatiques, de nettoyeurs de piscines et de robots mobiles comme le robot Pekee développé par Wany Robotics. Le projet exploratoire Wacif a consisté à réaliser ce type de démonstrateur afin d'étudier la faisabilité et l'utilité d'objets mobiles communicants au sein d'une telle maison.

La navigation autonome des robots mobiles peut également intervenir dans le contexte de l'aide aux personnes à mobilité réduite via la mise au point de fauteuils roulants autonomes facilitant les déplacements (comme dans [Nuttin 01] par exemple) ou la saisie d'objets (comme le robot SAM, développé au CEA-List dans le cadre du projet ANSO[4]).

De nouvelles applications pour les robots mobiles sont apparues avec les engins aériens autonomes. Dans [Sarris 01], huit domaines d'application civils sont proposés : surveillance des frontières, recherche et secours en cas d'accident ou de désastres, détection des feux de forêts, relais de communication, application des lois, gestion des désastres et des urgences (surveillance), plateforme pour la recherche scientifique (étude de l'environnement, de l'atmosphère) et applications industrielles (surveillance de pipelines, surveillance de centrales nucléaires) et agricoles (traitement de cultures). À cette liste, on peut ajouter la reconstruction 3D de sites archéologiques ou de villes. Ces applications nécessitent le développement de stratégies de navigation adaptées qui font l'objet

[3] Liselec : http ://www.liselec.fr/http ://www.liselec.fr/

[4] ANSO : http ://www.approche-asso.com/page.php ?pagegroup=25&pagecontenu=77http ://www.approche-asso.com/page.php ?pagegroup=25&pagecontenu=77

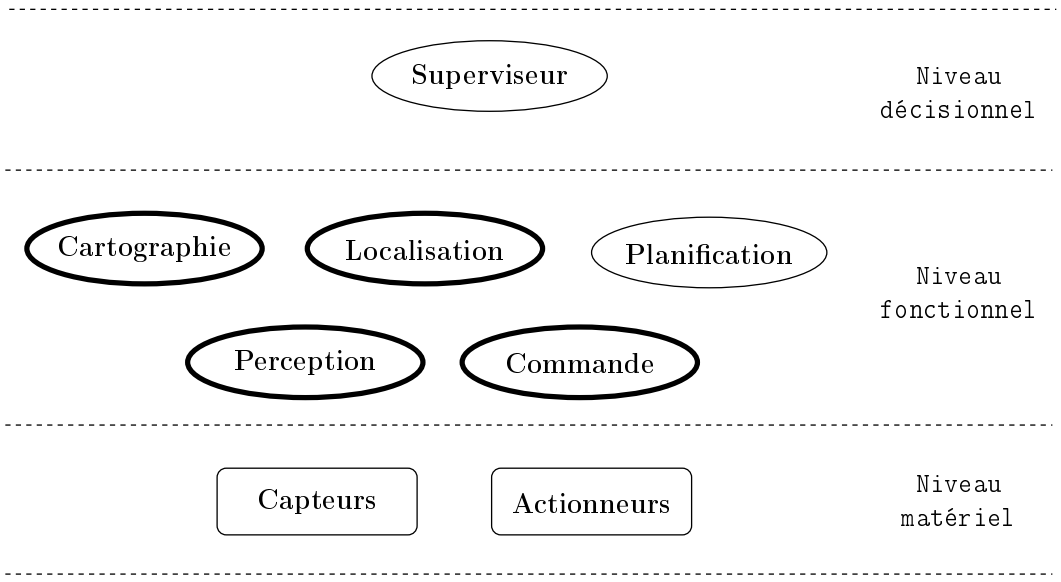


FIG. 1.4 – Éléments d'un système de navigation.

de nombreux travaux aux États-Unis et en Europe. Ainsi, le projet européen muFly [5] a pour ambition de développer un petit hélicoptère autonome de taille et de poids comparables à ceux d'un oiseau. Le projet européen μDrones[6] a pour objectif de développer de nouveaux concepts afin de rendre les robots volants autonomes. Le projet AVCAAF [7] mené à l'université de Floride en collaboration avec d'autres instituts américains étudie plusieurs scénarios : la détection et la cartographie de zones contaminées, la surveillance et le suivi ainsi que l'évaluation de dégâts suite à un conflit.

Ces applications montrent la diversité et la complexité associées à une tâche de navigation de robots mobiles. Cette tâche nécessite que le robot dispose à la fois de moyens de perception et d'un modèle de l'environnement afin que sa localisation puisse être effectuée et que ses déplacements puissent être planifiés et exécutés. Le système complet impliqué et ses éléments sont représentés Fig. 1.4. Ce système, inspiré de l'architecture LAAS [Alami 98], comporte trois niveaux. Le premier niveau (niveau décisionnel) inclue un superviseur gérant les modules du niveau fonctionnel. Ces modules peuvent être regroupés en plusieurs éléments principaux : cartographie, localisation, perception, planification et commande. Chaque module intègre un ensemble de fonctions reliées au niveau matériel. Ce dernier niveau contient les capteurs ainsi que les actionneurs. Nous détaillons dans la suite de cette section les différents éléments du niveau fonctionnel, les niveaux décisionnel et matériel n'ayant pas été traités dans nos travaux.

[5] muFly : http ://mufly.ethz.ch/http ://mufly.ethz.ch/
[6] μDrones : http ://www.ist-microdrones.orghttp ://www.ist-microdrones.org
[7] AVCAAF : *Active Vision for Control of Agile, Autonomous Flight*

1.1.1 Perception

Afin de réaliser une tâche de navigation, le robot doit percevoir son déplacement ainsi que l'environnement qui l'entoure. Pour cela, il peut être équipé de deux types de capteurs : les capteurs proprioceptifs (ou idiothétiques) et les capteurs extéroceptifs (ou allothétiques). Les capteurs du premier type donnent une information sur l'état interne du robot tandis que ceux du second type permettent de mesurer l'état du robot par rapport à son environnement.

Capteurs proprioceptifs Le déplacement effectué par le robot dans son environnement peut être estimé à partir de l'intégration des données sur l'état interne fournies par ces capteurs (vitesse, accélération ou rotations des roues par exemple). Le principal inconvénient des capteurs proprioceptifs provient de l'accumulation des erreurs lors de ces intégrations successives qui se traduit par une dérive de l'estimation du déplacement. Ces erreurs peuvent être classées en deux catégories : les erreurs systématiques proviennent des erreurs de modélisation et de mesure tandis que les erreurs non systématiques sont aléatoires. Les erreurs systématiques peuvent être modélisées afin d'être prises en compte ce qui est difficilement possible pour les erreurs de la seconde catégorie.

Les capteurs proprioceptifs les plus utilisés dans le domaine de la navigation de robots mobiles sont les odomètres pour les véhicules à roues et les centrales inertielles dans le cas des engins aériens ou sous-marins.

Les odomètres mesurent les vitesses de rotation des roues du véhicule. Le déplacement peut alors être estimé à partir de celles-ci et du modèle de déplacement du robot. Cette estimation subit des dérives importantes en raison des inévitables erreurs de modèle (erreurs sur le diamètre des roues, sur leur alignement, sur la longueur de l'entraxe ...), de mesure (résolution des encodeurs, fréquence d'acquisition inexacte ...) et aux hypothèses souvent simplificatrices sur la dynamique (glissement des roues sur le sol par exemple). Pour cette raison, l'odométrie est généralement utilisée uniquement pour estimer localement le déplacement.

Une centrale inertielle est composée de trois accéléromètres et de trois gyromètres. Les accéléromètres mesurent les accélérations en translation tandis que les gyromètres fournissent les vitesses en rotation. Il est nécessaire de filtrer les données fournies par ces capteurs car elles sont très bruitées. Différents outils sont utilisés à cette fin comme les filtres de Kalman, les filtres à particule ou le filtre complémentaire non-linéaire [Hamel 06b]. En pratique, l'orientation peut être estimée de façon correcte tandis que la vitesse de translation diverge rapidement ce qui rend l'estimation de la position inutilisable.

Capteurs extéroceptifs Afin de limiter la dérive des données acquises par les capteurs proprioceptifs, une solution consiste à recaler ces informations à partir d'autres données comme celles fournies par les capteurs extéroceptifs. En effet, les données acquises par les capteurs extéroceptifs ou *images* sont, elles, traitées sans intégration afin d'obtenir l'état du robot par rapport à son environnement. Ces capteurs fournissent donc

des observations avec une qualité constante dans le temps. Cependant, ils souffrent du problème d'ambiguïté perceptuelle (pour deux lieux différents de l'environnement, le contenu des images acquises par le capteur peut être similaire). Il est alors impossible d'estimer la localisation du robot sans avoir recours à des informations complémentaires. En outre, comme une image est une représentation de l'environnement tel qu'il est perçu par le capteur à un instant donné pour une prise de vue donnée, une image est sujette à la variabilité perceptuelle due aux variations de l'environnement au cours du temps. Ces variations proviennent des changements d'apparence (changement d'illumination par exemple) et des changements de contenu (déplacement, apparition ou disparition d'éléments).

Les capteurs extéroceptifs les plus usités en robotique mobile sont les capteurs de positionnement par satellites (en milieu extérieur), les télémètres laser et les caméras.

Les systèmes de positionnement par satellites comme le GPS (*Global Positionning System*) ou Galileo sont très intéressants car ils permettent d'obtenir par un processus de triangulation un positionnement géo-référencé. Une précision de quelques centimètres peut être atteinte à l'aide d'un GPS Différentiel cinématique temps réel (RTK-DGPS pour *Real Time Kinematic Differential GPS*) tandis que des GPS à bas coûts ont une précision de l'ordre de 10 mètres. Il est à noter que l'utilisation des GPS requière une bonne couverture satellitaire. Cette condition dépend entre autres du type d'environnement et n'est pas remplie dans les environnements d'intérieur et dans les *canyons* urbains (routes encadrées de hauts bâtiments). De plus, en attendant l'arrivée de Gallileo, le système GPS, propriété de l'armée américaine, n'est pas assuré et peut être interrompu à tout moment.

Un capteur largement utilisé dans le cadre de la navigation de robots mobiles à roues navigant dans des environnements structurés est le télémètre laser. La carte de profondeur obtenue par le télémètre peut être utilisée dans les approches de localisation et cartographie simultanées ainsi que pour la détection d'obstacles. Les capteurs employés sont généralement des télémètres laser 2D, situés dans le plan horizontal mais on trouve également des télémètres multi-couches qui permettent d'acquérir des données sur plusieurs plans 2D. Les télémètres 3D apparaissent dans des applications spécifiques telles que les systèmes de cartographie mobile mais nécessitent de gros moyens de stockage et de traitement des données.

Les caméras sont également largement utilisées. Un état de l'art sur les stratégies de navigation de robots mobiles basées sur un capteur de vision jusqu'en 2002 est proposé dans [DeSouza 02]. De nombreux ateliers (comme, par exemple, [Wor 08]) ainsi que de nombreux numéros spéciaux dans des revues (comme, par exemple, [TRO 08]) traitent de ce sujet. Avec l'augmentation des capacités de calculs des ordinateurs, le traitement des images est devenu suffisamment rapide et leur utilisation pour la navigation des robots a donc cru. Nos travaux exploitant largement ce type de capteur, nous reviendrons plus en détails sur ceux-ci dans la suite de ce manuscrit.

1.1.2 Cartographie

La carte est un support de navigation pouvant contenir des informations de types très différents. Les principaux types de cartes se retrouvent dans l'approche de hiérarchie sémantique spatiale[8] développée dans [Kuipers 00] : la carte sensorielle, la carte commande, la carte causale, la carte topologique et la carte métrique.

Une carte sensorielle contient des informations proches des données acquises par les capteurs. Il peut s'agir, par exemple, des images visuelles ou de grilles d'occupation locales construites à partir des distances mesurées par les télémètres 2D. Étant donné que ce type de carte contient un ensemble d'images, on parle également de *mémoire* en référence aux approches biologiques (voir Section 1.2.1). Une carte commande décrit l'environnement avec des segments. Ces segments associent aux données capteurs les actions envoyées aux actionneurs. Une carte causale est un modèle abstrait discret contenant des schémas d'action. Un schéma est un triplet $< \mathcal{I}, A, \mathcal{I}' >$ qui définit le passage de l'image $\mathcal{I}$ à l'image $\mathcal{I}'$ via l'action A. Une carte topologique permet de définir les lieux, chemins et régions de l'environnement ainsi que leur connectivité. Enfin, une carte métrique permet de définir la géométrie globale de l'environnement dans un repère de référence.

Les représentations de l'environnement utilisées dans la littérature exploitent une ou plusieurs de ces cartes. La plus simple consiste à associer une carte sensorielle contenant un élément unique (un objet visible [Braitenberg 84] ou la configuration spatiale d'*amers* [9] [Cartwright 83, Gourichon 04]) à une carte commande contenant la commande à utiliser pour atteindre cet élément (voir Figure 1.5 (a)). Cette représentation est locale au sens où elle ne peut être utilisée que si l'objet à atteindre ou les amers sont visibles.

Les cartes sensorielle et commande peuvent également être exploitées conjointement afin de rejoindre un objectif lointain. Pour cela, une séquence d'informations extraites de la carte sensorielle associées aux actions extraites de la carte commande, appelée *route* dans la littérature, peut être utilisée comme proposé dans [Matsumoto 96, Gaussier 97, Giovannangeli 06] (voir Fig. 1.5 (c)). Ces stratégies permettent une autonomie plus importante mais sont limitées à un objectif unique. Afin de définir un nouvel objectif, une nouvelle route doit être apprise. Afin de surmonter cette difficulté, les cartes métrique et topologique contiennent des réseaux de routes. Cette dernière approche est très intéressante car les possibilités d'adaptation et d'optimisation face aux changements de l'environnement sont importantes. Ces deux types de cartes permettent de modéliser des environnements de grande taille ainsi que de planifier des trajets vers un objectif lointain mais ils sont, en contrepartie, plus complexes à mettre en œuvre.

[8] *Spatial Semantic Hierarchy (SSH)*

[9] Un amer est un point distant remarquable (fixe), naturel ou artificiel, permettant de se repérer. Ce terme est emprunté au domaine maritime.

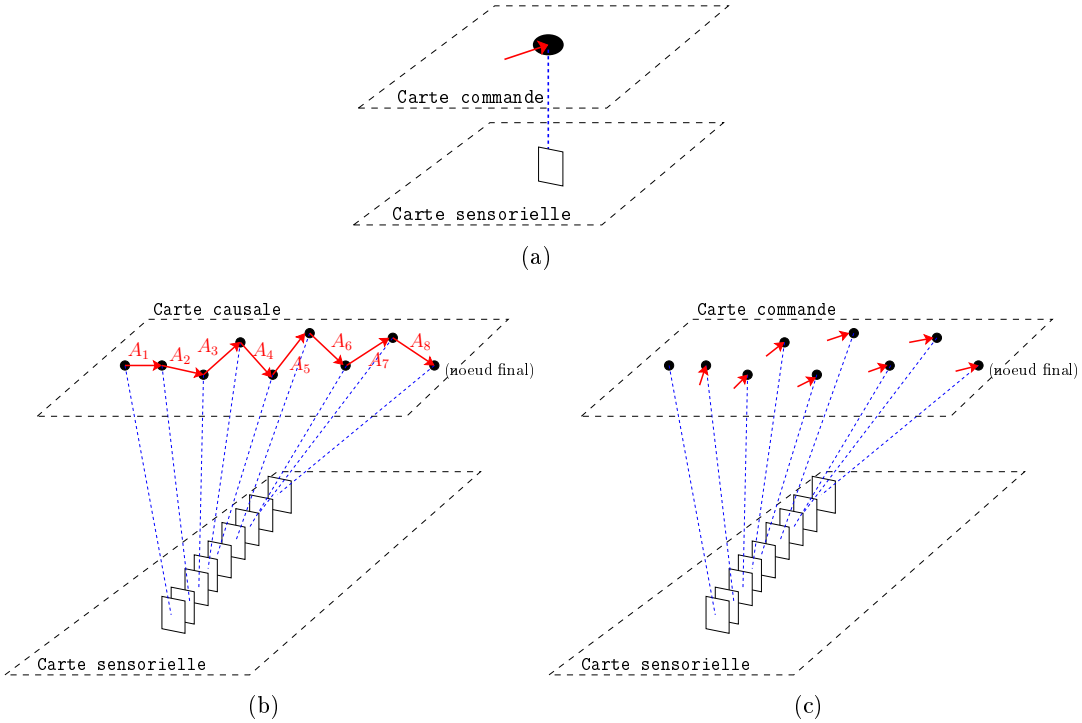
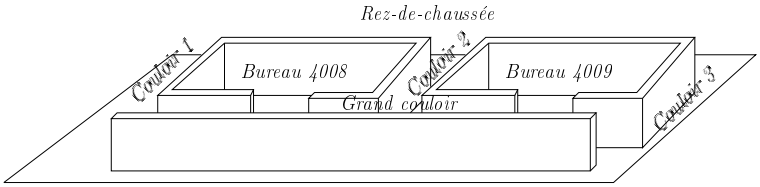
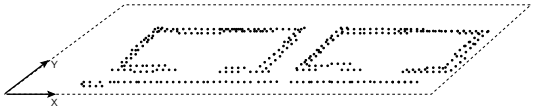


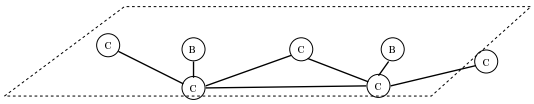
FIG. 1.5 – Représentation sous la forme (a) d'une carte simple, (b) d'une route explicite et (c) d'une route implicite. Dans le premier cas, l'objectif est de rejoindre l'image en utilisant la loi de commande. Un autre objectif consiste à suivre une route, c'est-à-dire à se déplacer d'une image à une autre. Cette méthode emploie alors, en plus de la carte sensorielle, une carte commande ou une carte causale. Dans le premier cas, la représentation assure que l'image $\mathcal{I}'$ est obtenue si on était à $\mathcal{I}$ et que l'action A est effectuée, ce qui n'est pas le cas de la seconde représentation.



(a) Environnement réel d'intérieur



(b) Carte métrique 2D



(c) Carte topologique

FIG. 1.6 – Représentation de l'environnement réel sous la forme d'un modèle géométrique (b) en 2D avec des points et (c) sous la forme d'une carte topologique avec des lieux caractéristiques.

Dans les cartes métriques, la structure de l'environnement est décrite dans un repère absolu [Chatila 85]. Une première approche consiste à représenter l'environnement par un ensemble de primitives géométriques auxquelles sont associées une position et une orientation définies dans ce repère (voir Fig. 1.6 (b)). Une approche alternative consiste à représenter l'espace libre et non les primitives géométriques. L'approche la plus utilisée dans ce contexte est basée sur la notion de grille d'occupation. L'environnement est alors discrétisé en cellules auxquelles sont associées une probabilité d'occupation [Elfes 89]. En pratique, cette dernière représentation se heurte aux volumes importants de données à mémoriser, ce qui rend son utilisation plus adaptée en 2D et très difficile dans des environnements de grande taille.

Les cartes topologiques permettent une représentation discrète de l'environnement sous la forme d'un graphe [Kuipers 91, Choset 01, Thrun 02] (voir Fig. 1.6 (c)). Les noeuds de ce graphe représentent généralement les positions atteignables par le robot tandis que les arêtes traduisent la navigabilité d'un lieu à un autre. Une autre possibilité consiste à définir un noeud comme une transition entre deux lieux [Kortenkamp 94]. Dans ce contexte, les noeuds d'un environnement d'intérieur correspondent à des portes ou à des intersections de couloirs par exemple. De façon générique, un noeud peut être étiqueté parmi un ensemble de possibilités. Un noeud peut également contenir des informations proches des données capteurs brutes (carte sensorielle) comme les signatures télémétriques [Kuipers 91, Matarić 92], les images visuelles [Kortenkamp 94, Santos-Victor 99] ou les grilles d'occupation locales [Yamauchi 97]. Quant aux arêtes, elles peuvent aussi contenir des informations métriques comme la longueur à parcourir entre deux lieux.

Diverses fonctionnalités sont nécessaires afin de mener à bien une tâche de navigation (localisation, planification, commande pour ne citer que celles que nous décrivons dans ce document). Il est de notre point de vue souhaitable d'utiliser conjointement plusieurs types de cartes afin de supporter l'ensemble de ces fonctionnalités. Ces cartes peuvent être hétérogènes. Un exemple typique est la combinaison des cartes topologiques et métriques [Tomatis 01]. Une telle représentation correspond, par exemple, à une carte topologique dont les noeuds sont des cartes métriques locales [Simhon 98, Buschka 04] (voir Fig. 1.7). Depuis quelques années, les Systèmes d'Information Géographique (SIG) font leur apparition. Il s'agit d'outils informatiques permettant d'organiser et de représenter des données référencées spatialement (cartes métriques). Un SIG peut être employé pour la navigation en milieu urbain comme il est proposé dans [Bonnifait 08]. Il peut être composé de cartes contenant les supports de navigation, les amers, des images, la traversabilité et un modèle numérique de terrain (représentation proposée au LAAS). Il est également envisageable de décomposer une carte hiérarchiquement en plusieurs niveaux de détails.

Comme on le voit avec l'étiquetage des noeuds des cartes topologiques, il est intéressant d'ajouter aux cartes spatiales des concepts abstraits permettant de donner un sens aux éléments constituant la carte. Dans [Galindo 05], une hiérarchie spatiale

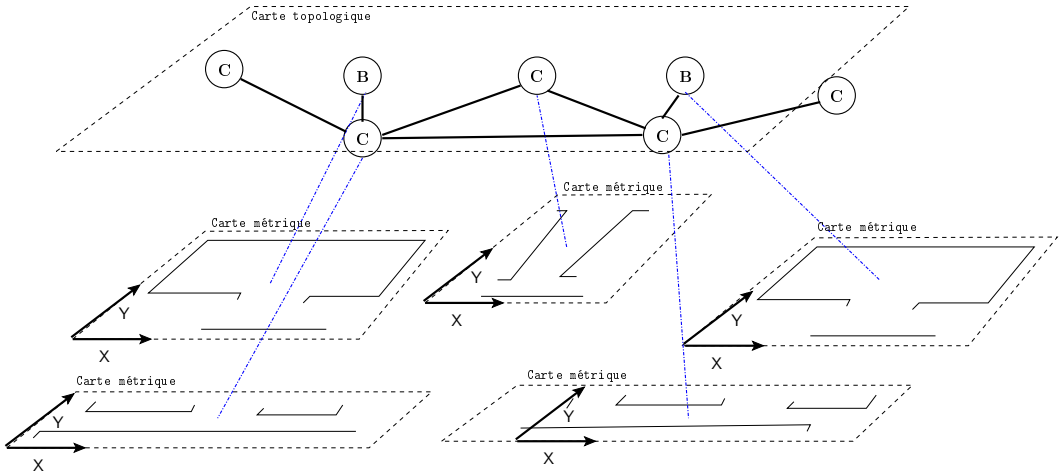


FIG. 1.7 – Représentation de l'environnement réel sous la forme d'une carte hybride métrique-topologique où des cartes métriques locales sont associées aux noeuds d'une carte topologique.

est définie avec une carte sensorielle et une carte topologique à deux niveaux. À cette représentation spatiale est ajoutée une seconde représentation hiérarchique modélisant les connaissances sémantiques sur l'environnement et utilisant une ontologie c'est-à-dire un modèle de données représentatif d'un ensemble de concepts et des relations les liant (voir Fig. 1.8). Les éléments des hiérarchies spatiale et sémantique sont ensuite reliés par des ancrages entre les composantes de chaque hiérarchie. La hiérarchie sémantique permet d'avoir une représentation de haut niveau et de réaliser l'interface entre le robot et les utilisateurs. Elle peut également être utilisée afin de détecter des erreurs de localisation en raisonnant à partir de la reconnaissance de lieux ou d'objets. Dans [Rottmann 05], une approche supervisée permet, par exemple, de classer les différentes pièces d'un environnement d'intérieur (laboratoire, couloir, porte, cuisine ou bureau) à partir de données visuelles et télémétriques.

Construction de la carte La construction des cartes métriques se base sur l'appariement des données courantes aux données déjà présentes dans la carte. Cette construction dépend donc fortement de la précision des capteurs utilisés. Dans le contexte des capteurs visuels, la technique de reconstruction 3D à partir d'une caméra en mouvement (désignée en anglais par "Sfm" pour *Structure from motion*) consiste à estimer la structure 3D en analysant le mouvement de l'objet. Il est ensuite possible de minimiser l'erreur de reprojection des points sur les images en faisant varier la structure de la scène c'est-à-dire les poses des caméras depuis lesquelles les images ont été acquises. Cette phase, appelée ajustement de faisceaux, a été exploitée entre autres dans [Nistér 06, Royer 07, Tardif 08]. Dans les approches par filtrage ou les approches bayési-

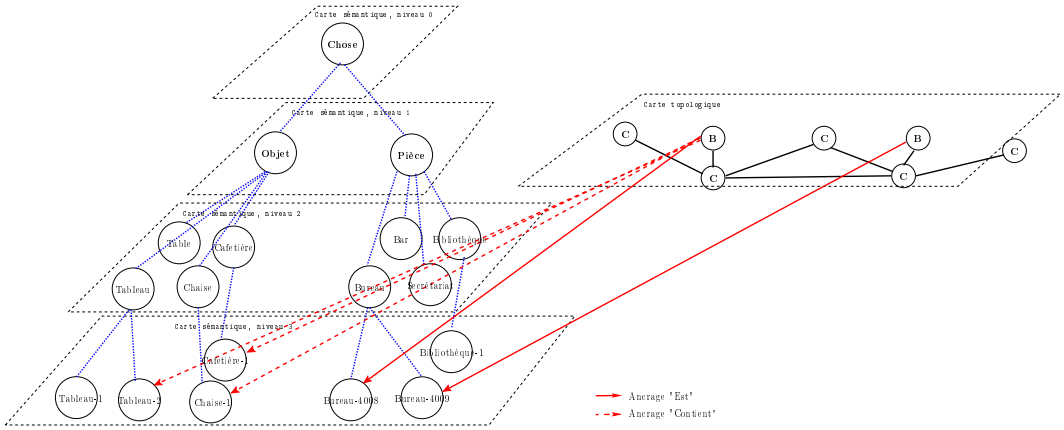


FIG. 1.8 – Carte sémantique et carte spatiale topologique.

ennes (dont le SLAM[10]), la structure et la localisation sont estimées simultanément en tenant compte d'un modèle de déplacement. Le lecteur pourra se référer aux tutoriaux de Bailey et Durrant-Whyte [Durrant-Whyte 06, Bailey 06] pour les méthodes les plus classiques de SLAM. La plupart des travaux sur le SLAM utilisent un télémètre laser tandis que les premiers résultats dans le domaine du SLAM visuel ont été présentés dans [Davison 03]. Dans ce contexte, les travaux exploitent une caméra unique et des primitives de type points d'intérêts ([Davison 03, Solà i Ortega 05]), des systèmes de stéréovision ([Lemaire 07]) ou des approches d'appariement dense ([Silveira 08]).

Lorsque la cartographie porte sur des environnements de grande échelle, il est souhaitable d'intégrer à celle-ci des informations concernant les éventuels retours du robot à une situation déjà rencontrée. La détection et l'intégration de ces événements au processus de cartographie est un problème important et est largement abordé dans la littérature sous le terme de fermeture de boucle. Celle-ci peut permettre de corriger les erreurs d'estimation accumulées comme dans [Victorino 05]. Le lecteur pourra se référer entre autres à [Beevers 05, Angeli 08a] pour plus d'informations sur cette problématique.

1.1.3 Localisation

L'étape de localisation consiste à déterminer la situation du robot dans sa carte. Les approches de localisation peuvent être divisées en deux catégories [Filliat 01, Filliat 03] :

– *l'inférence directe de position* : seules les données proprioceptives courantes sont dans ce cas utilisées pour localiser le robot. Dans cette approche, on fait les hypothèses que l'environnement n'est pas confronté à des problèmes d'ambiguïté perceptuelle et que les algorithmes utilisés sont robustes aux bruits de mesure et aux changements de contenu.

[10]SLAM : *Simultaneous Localization And Mapping*, localisation et cartographie simultanée

- *le suivi de plusieurs hypothèses de position* : les données proprioceptives et extéroceptives présentes et passées sont dans ce cas utilisées pour alimenter le processus de localisation. Plusieurs alternatives peuvent être conservées en mémoire et confrontées en utilisant :
  - des hypothèses explicites [Arleo 00],
  - une distribution de probabilités de présence sur l'ensemble des positions. Cette distribution permet de prendre en compte les informations proprioceptives (déplacement des probabilités) ainsi que les informations extéroceptives (modulation des probabilités). A. Angeli utilise ce procédé dans [Angeli 08a] afin de traiter du problème de fermeture de boucle.

1.1.4 Planification

Dans le cas d'une carte topologique, la phase de planification consiste à définir une séquence de noeuds amenant le système robotique dans une situation souhaitée à partir de la situation initiale tandis que dans le cas d'une carte métrique, elle consiste à définir une séquence discrète ou continue de situations définies dans le repère absolu. On pourra se référer à l'ouvrage [Latombe 91] pour les techniques de planification dans une carte métrique. Dans ce contexte, l'approche la plus classique consiste tout d'abord à discrétiser l'espace libre : l'environnement est décomposé en cellules ou en chemins simples. Parmi les méthodes de construction de chemins, on peut citer le diagramme de Voronoï [Choset 95], l'extraction de silhouettes, le graphe de visibilité [Latombe 91]. La planification s'apparente alors, comme dans le cas des cartes topologiques, à la recherche dans un graphe. Les algorithmes utilisés peuvent être les algorithmes de Bellman–Ford ou de Dijkstra (voir, par exemple, [Cherkassky 96]).

1.1.5 Commande

Les objectifs de la commande peuvent être divers : suivre un chemin, éviter les obstacles, garder un objet dans le champ de vue du capteur ... Plusieurs architectures de commande sont possibles pour réaliser ces objectifs.

Parmi celles-ci, on peut citer les architectures délibératives qui consistent en un découpage modulaire des fonctionnalités nécessaires à la navigation. Les modules sont agencés verticalement et reliés en série (voir Figure 1.9 (a)). On parle alors également de schéma perception-planification-action[11]. Dans ce schéma, la couche avec la hiérarchie la plus haute est en contact direct avec les capteurs et met en forme les grandeurs physiques mesurées pour les rendre utilisables par la couche de modélisation. Ces informations sont transformées et permettent de mettre à jour une carte interne de l'environnement. À partir de cette carte, la couche de planification met en place les actions et la dernière couche est responsable de leur exécution. Une telle approche est notamment proposée

[11] *Sense-plan-act*, littéralement "détecter, planifier, agir"

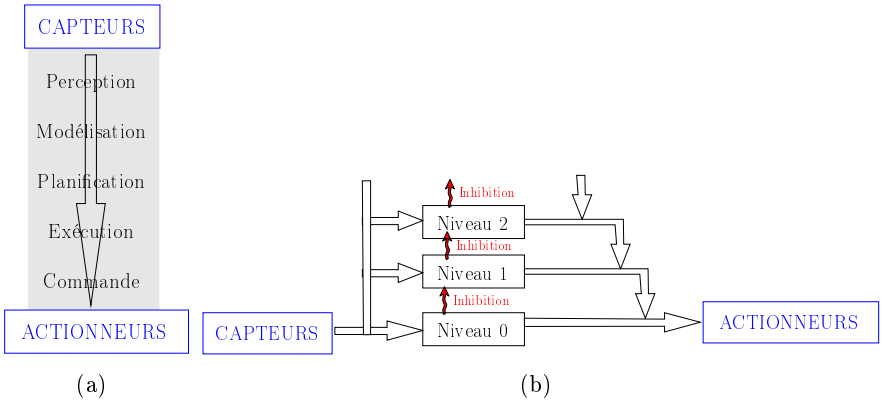


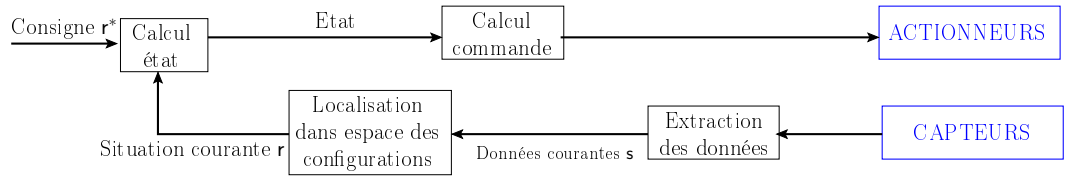
FIG. 1.9 – Architecture de commande (a) délibérative, (b) à *subsomption*.

dans [Novales 94].

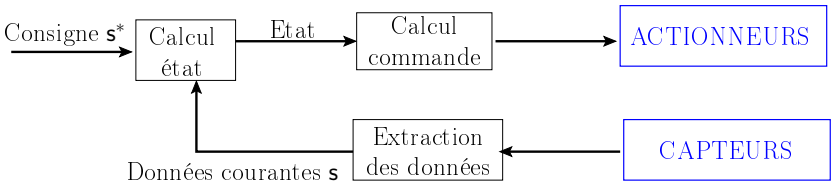
Un inconvénient des architectures délibératives est qu'elles ne prennent pas en compte les changements dynamiques de l'environnement dans la carte. Afin de résoudre ce problème, des architectures dites *réactives* emploient directement les informations acquises par les capteurs sans passer par une utilisation ou mise à jour de la carte. Dans l'architecture réactive à *subsomption*, proposée dans [Brooks 86], les fonctions sont distribuées sur plusieurs niveaux travaillant en parallèle (voir Fig. 1.9 (b)). Ce découpage horizontal de type perception-action s'oppose ainsi au découpage vertical vu précédemment. Chaque niveau utilise les données issues des capteurs, et actionne directement les effecteurs. Ces niveaux sont gérés automatiquement par priorité, un module inférieur prenant la priorité sur un module supérieur s'il est activé.

D'un côté, les architectures délibératives permettent de construire une représentation de l'environnement mais elles débouchent souvent sur une imbrication forte entre les différentes fonctionnalités. De ce fait, l'identification des interconnexions à mettre en place peut s'avérer problématique. D'un autre côté, les différents niveaux d'une architecture réactive sont simples à définir et sont relativement indépendants mais elles rendent difficile l'utilisation des modèles définis dans un niveau différent. Il est de plus nécessaire de définir les priorités entre les niveaux. D'autres architectures ont été développées comme le mécanisme de coordination d'actions [Pirjanian 99], la sélection d'actions par réseau d'activation [Maes 90] ou l'architecture orientée schéma [Arbib 81].

Les lois de commande implémentées dans ces architectures peuvent utiliser différents concepts. Les lois de commande peuvent classiquement être définies dans l'espace des configurations. L'état du robot est alors estimé à partir de sa localisation dans un modèle global de l'environnement (voir Fig. 1.10 (a)). Afin d'éliminer la phase de localisation globale, il est possible d'exploiter les concepts relatifs à la commande référencée capteur. Dans ce cas, les tâches à réaliser sont exprimées non plus dans l'espace des con-



(a) Commande classique



(b) Commande référencée capteurs

FIG. 1.10 – Commande classique et commande référencée capteurs. r représente la situation dans l'espace des configurations et s le vecteur de données capteurs.

figurations mais directement dans l'espace du capteur (voir Fig. 1.10 (b)). De ce fait, cette catégorie de commande est réputée plus robuste vis-à-vis des bruits de mesure et des erreurs de modélisation que les commandes appartenant à la catégorie précédente. Lorsque le capteur est une caméra, on parle alors d'asservissement visuel (voir, par exemple, les tutoriaux [Hutchinson 96, Cervera 04, Chaumette 06, Chaumette 07]). Depuis de nombreuses années, cette approche est appliquée avec succès notamment sur les robots manipulateurs.

1.2 Approche cognitive

Dans la majorité des cas, l'homme a façonné l'environnement dans lequel navigue un robot. Il se localise, planifie ses déplacements et se déplace dans ces environnements en mettant en correspondance sa perception courante de la scène réelle observée avec un modèle interne constitué au fur et à mesure de ses expériences de navigation. Pour cela, il dispose d'un système complet de traitement et de mémorisation des données. Dans cette Section, nous nous intéressons à ce système souvent source d'inspiration des développements en robotique.

Nous présentons tout d'abord la représentation de l'espace employée par l'homme ainsi que ses comportements lors des déplacements (voir Section 1.2.1). Cette représentation est contenue dans la mémoire (terme défini en psychologie comme la faculté de l'esprit de stocker, conserver et rappeler des informations et des expériences passées) qui peut être décomposée en fonction du type d'information mémorisé et de leur durée de rétention (voir Section 1.2.2).

1.2.1 Carte cognitive et navigation

D'après [Tolman 48], un individu possède une représentation mentale de l'espace dans lequel il se trouve appelée *carte cognitive*. Cette carte représente différents espaces décrits relativement à soi [Tversky 01] : l'espace du corps, l'espace autour du corps, l'espace de navigation et l'espace abstrait. L'espace du corps contient les données relatives à l'état interne (données idiothétiques) obtenues via un modèle corporel. L'espace autour du corps contient les éléments de l'environnement qui peuvent être vus et rejoints depuis la position courante. L'espace de navigation est l'espace dans lequel l'homme peut se déplacer. Il en possède une construction mentale représentée sous la forme d'un schéma simplifié. Enfin, l'espace abstrait ou espace des graphiques contient des représentations graphiques telles que des cartes, des schémas, des dessins et des diagrammes.

L'organisation de l'espace de navigation nous intéresse ici plus particulièrement. D'après [Golledge 01], cet espace est organisé sous la forme d'une base de données indexée. Il n'est pas sûr que cette représentation soit enregistrée comme une carte spatiale mais l'analyse des cellules de lieux suggère que les lieux différents sont mémorisés dans différentes parties du cerveau [O'Keefe 78]. Des expérimentations sur des rats ont montré qu'un système d'intégration du mouvement n'est pas nécessaire pour se localiser dans l'espace de navigation [Redish 99]. En outre, l'estimation de la position métrique et de l'orientation n'a qu'un rôle de support dans ce processus d'après [Kuipers 91].

Cette représentation de l'espace de navigation est employée pour réaliser des déplacements. Chez les hommes, trois comportements spatiaux ressortent des études : les personnes peuvent avoir une connaissance des *amers*, des *routes* ou des *configurations* [Golledge 01]. Dans le premier cas, elles sont capables de reconnaître un lieu mais ne peuvent planifier un chemin entre deux noeuds. Elles doivent alors faire appel à d'autres sources d'informations pour la planification (demander à d'autres personnes, rechercher sur une carte extérieure ...). Dans le deuxième cas, les personnes mémorisent les lieux mais également les chemins pour aller d'un lieu au suivant. Elles sont donc capables de suivre une route apprise. Par contre, elles ne sont pas capables de changer de route une fois que celle-ci a été planifiée, ni de prendre des raccourcis. Enfin, les personnes de la dernière catégorie peuvent développer de nouvelles routes à partir de la connaissance de la nature du réseau de routes. Cette dernière approche est certainement la plus intéressante car les possibilités d'adaptation et d'optimisation face aux changements de l'environnement sont très importantes.

On s'aperçoit que l'homme n'utilise pas uniquement une représentation spatiale de l'environnement pour se déplacer. Il emploie également des schémas généraux et des gabarits de lieux acquis lors d'expériences passées. Même dans un lieu où il n'est jamais venu, un homme est capable de retrouver des repères. Il peut ainsi se repérer facilement dans un magasin ou dans un réseau de transport car il a déjà fait face à ce genre de situation auparavant. D'autre part, le comportement de l'homme est également influencé par les habitudes. Deux personnes peuvent ainsi prendre des chemins totalement

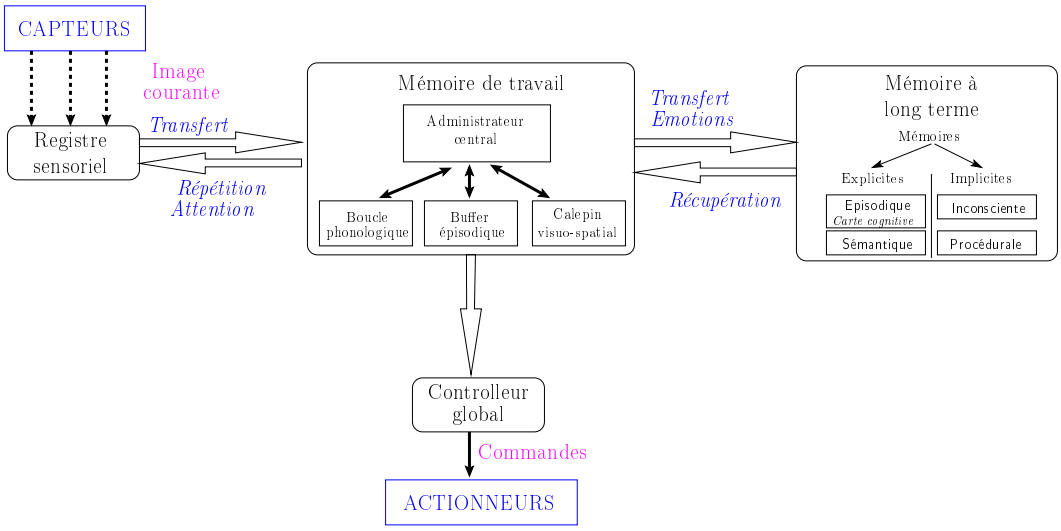


FIG. 1.11 – Représentation schématique du modèle de la mémoire.

différents pour se déplacer entre deux mêmes lieux.

1.2.2 Organisation de la mémoire

La représentation de l'espace de navigation est obtenue à partir de phases d'apprentissage où les éléments utiles sont mémorisés. D'après le modèle formulé dans [Atkinson 68], la mémoire peut être décomposée en fonction du type d'information mémorisé et de la durée de rétention. Dans ce modèle qui a émergé dans la psychologie cognitive de la fin des années 1960, le système de mémorisation peut être décomposé en trois modules : le registre sensoriel, la mémoire à court terme (MCT) - mémoire de travail et la mémoire à long terme (voir Figure 1.11). Concernant la navigation, la mémoire à long terme (MLT) contiendrait entre autres la carte cognitive, les expériences passées ainsi que les habitudes.

Registre sensoriel Le registre sensoriel réalise un pré-traitement des informations reçues par les capteurs (ou *stimuli*) et les code. L'information du stimulus sensoriel est conservée pendant une durée dépassant sa durée de présentation. Les stimuli les plus importants concernent les informations visuelles (conservées durant environ 300 millisecondes) et les informations auditives (conservées pendant une à deux secondes). Lorsque l'être humain exécute une action, il focalise son attention sur certains éléments, ce qui influence les pré-traitements réalisés par le registre sensoriel.

Mémoire à court terme et mémoire de travail La mémoire à court terme est caractérisée par une capacité limitée de 5 à 7 items (unités d'information pouvant être

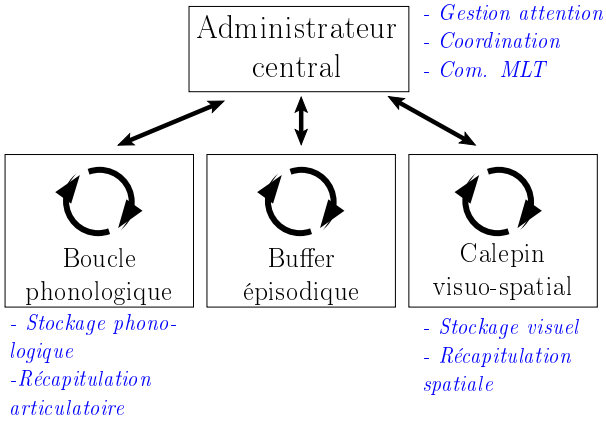


FIG. 1.12 – Composantes de la mémoire de travail.

retenues dans l'empan mnésique) [Miller 56] et par une durée de conservation de ces informations relativement courte (de l'ordre d'une dizaine de secondes). C'est dans la mémoire de travail que les informations stockées dans la MCT en prévision d'une action sont manipulées. Dans la suite, nous inclurons la mémoire de travail dans la MCT.

Plusieurs représentations de la mémoire de travail ont été proposées dans la littérature comme, par exemple, le modèle des processus emboîtés [Cowan 99] et le modèle à composantes multiples [Baddeley 74]. Nous nous focaliserons ici sur le modèle à composantes multiples, communément employé. Dans cette représentation, la MCT est composée d'un administrateur central et de deux systèmes esclaves : la boucle phonologique et le calepin visuo-spatial [Baddeley 74] (voir Fig. 1.12). La boucle phonologique est destinée au stockage temporaire de l'information verbale. Le mécanisme de récapitulation articulatoire permet de rafraîchir cette information et de transférer l'information verbale présentée visuellement. Le système responsable des informations visuo-spatiales est impliqué dans la génération et la manipulation des images mentales. Enfin, l'administrateur central ou centre exécutif est responsable du maintien temporaire des informations et utilise pour cela les deux systèmes esclaves. Dans le modèle de Luria [Luria 85], les fonctions d'exécution principales de l'administrateur sont l'identification et le maintien d'un objectif, la planification, l'exécution du programme planifié et la vérification du résultat. Le buffer épisodique peut être associé à ces trois modules. Il est dédié au stockage temporaire d'informations multimodales et à l'intégration des informations provenant des autres systèmes esclaves et de la mémoire à long terme.

Mémoire à long terme Le processus de transfert des données de la mémoire de travail vers la MLT passe par une phase de codage des informations. Seules les informations les plus pertinentes et les plus significatives sont conservées et organisées. Ces informations peuvent ensuite être restituées lorsqu'il est nécessaire. Cette mémoire a une capacité *a priori* infinie et son contenu varie peu avec le temps.

Mémoires			
	EXPLICITE	IMPLICITE	
Expériences personnelles	Épisodique	Inconsciente	*Mesures indirectes d'expériences passées*
Faits et connaissances générales	Sémantique	Procédurale	*Habiletés motrices et savoir-faire*

TAB. 1.1 – Décomposition de la mémoire à long terme.

La MLT peut être décomposée en quatre catégories : les mémoires épisodique, sémantique, inconsciente et procédurale [Tulving 85] (voir Tab. 1.1). Les deux premières catégories sont explicites et permettent de garder les événements liés à l'apprentissage ("savoir que") tandis que les deux autres catégories sont implicites et permettent d'apprendre sans retenir le souvenir de l'apprentissage ("savoir comment"). La mémoire épisodique contient les souvenirs d'événements vécus (expériences personnelles). Elle dépend donc du contexte dans lequel les informations ont été mémorisées. La représentation mentale de l'espace (la carte cognitive) est contenue dans cette mémoire. La mémoire sémantique porte sur les faits et les connaissances générales. Elle fonctionne par des concepts objectifs, ce qui la rend plus fiable et plus sûre que la mémoire épisodique. La mémoire inconsciente contient des mesures indirectes de la rétention d'expériences passées. Enfin, la mémoire procédurale porte sur les habiletés motrices, les savoir-faire et les gestes habituels. Elle permet ainsi de mémoriser l'exécution d'une séquence de gestes afin de réaliser une tâche donnée.

1.3 Cadre de travail

Dans le cadre de cette thèse, nous nous sommes intéressés à l'amélioration de l'autonomie et des capacités d'adaptation des systèmes robotiques et plus particulièrement aux stratégies de navigation référencée capteurs de robots mobiles dans des environnements a priori inconnus. La navigation adresse des problématiques complexes qu'il est difficile de découpler pour bâtir un système autonome complet. Nous nous sommes donc focalisés sur la définition d'une approche complète allant de l'apprentissage de site à la commande automatique du robot.

1.3.1 Motivations

Cette thèse a été proposée conjointement par le LAboratoire des Sciences et Matériaux pour l'Électronique et d'Automatique (LASMEA) et par le Laboratoire de Téléopération et Cobotique (LTC) du Commissariat à l'Énergie Atomique, Laboratoire d' Intégration des Systèmes et des Technologies (CEA-LIST) et a été réalisée pour moitié dans chaque laboratoire. Elle a été cofinancée par la Région Auvergne et le CEA. Les deux

laboratoires travaillent depuis de nombreuses années sur la problématique de la navigation autonome des robots, l'objectif étant d'augmenter l'autonomie du robot. Leurs cibles applicatives peuvent être classées en trois catégories :

- les robots à roues dans les maisons communicantes.
  Les deux laboratoires ont, par exemple, participé au projet exploratoire Wacif où ils ont étudié et développé un assistant robotique à roues réalisant des tâches de téléprésence et de surveillance dans un environnement d'intérieur.
- les véhicules urbains autonomes.
  Dans ce contexte, le LASMEA a participé à de nombreux projets dont MobiVIP[12]. Le LASMEA travaille actuellement sur la mise à disposition d'une flotte de véhicules totalement autonomes dans le cadre du projet VIPA (Véhicules Intelligents Publics Automatiques).
- les robots aériens.
  Lors du projet français RobVolInt (Robot Volant d'Intérieur) [Hamel 06a], le CEA a développé un drone quadrirotor. Afin de rendre la commande d'un tel drone plus abordable aux personnes novices, N. Guénard a réalisé lors de sa thèse des algorithmes de commande permettant une téléopération simple de ce type d'appareil [Guénard 06]. Les travaux actuels, dont ceux effectués dans le projet européen μDrones, portent sur la navigation autonome de ces engins.

1.3.2 Approche proposée

Notre objectif est de bâtir une approche de navigation complète mais également suffisamment souple et légère d'un point de vue algorithmique pour être intégrée sur une électronique dédiée destinée à équiper un robot autonome. La complexité des tâches de navigation résulte en particulier de l'étendue spatiale de l'environnement de travail du robot et de la nature dynamique de cet environnement. Cependant, cet espace contient généralement des éléments fixes permettant de le caractériser et dont l'observation peut s'avérer très utile à la localisation.

Dans les applications envisagées, les robots se déplacent dans des espaces urbains ou domestiques façonnés par l'homme. Comme nous l'avons vu dans le paragraphe 1.2.1, l'homme en connaît un modèle interne issu de processus complexes de mémorisation de données allothétiques et idiothétiques acquises au fur et à mesure de ces expériences de navigation. Il se localise et planifie ses déplacements en mettant en correspondance les modèles issus de ses expériences passées et sa perception de la scène réelle observée. De la même manière, le développement d'une stratégie de navigation pour les robots mobiles nécessite l'utilisation d'une représentation de l'environnement. Assez logiquement, cette représentation peut s'inspirer de la représentation de l'espace de navigation que l'on retrouve chez l'homme. Cette représentation doit, bien entendu, permettre au robot d'alimenter tous les processus nécessaires à sa navigation (voir Section 1.1) mais peut également être vue comme un moyen de communication entre l'utilisateur et le robot.

[12]MobiVIP : http ://www-sop.inria.fr/visa/mobivip/http ://www-sop.inria.fr/visa/mobivip/

L'utilisateur peut notamment spécifier un objectif de navigation au robot en désignant directement une situation à atteindre dans la représentation. Le robot est alors à même de proposer une visualisation de sa localisation dans cette même représentation.

Le succès des approches référencées capteurs pour le contrôle des mouvements des robots manipulateurs, s'affranchissant d'une localisation géométrique absolue du robot, encourage la transcription des méthodes formalisées dans ce cadre à la navigation des robots mobiles. Comme nous l'avons vu dans la Section 1.1.5, les approches référencées capteurs consistent à spécifier la tâche robotique dans l'espace d'observation du système de perception. Les informations sensorielles acquises continuellement par le robot alimentent directement une loi de commande destinée à stabiliser ces informations sensorielles sur une consigne de même nature. Les approches référencées capteurs doivent leur popularité à leur robustesse intrinsèque. Spécifiée à l'aide d'informations sensorielles, la tâche robotique ne nécessite pas de modèle complexe de la scène ni de processus d'estimation d'une pose absolue du robot, coûteuse et incertaine, pour être accomplie. Cependant, l'utilisation d'approches référencées capteurs en robotique mobile se heurte à deux problèmes majeurs :

- D'une part, le robot est sujet à de grands déplacements, ce qui induit que les informations sensorielles de consigne ne peuvent pas forcément être mises en correspondance avec les informations sensorielles courantes.
- D'autre part, les robots mobiles se caractérisent en grande majorité par des contraintes de déplacement fortes (non-holonomie, sous actionnement, ...). La nature non linéaire de ces contraintes impose alors une remise en cause des approches de commande communément utilisées en robotique manipulatrice.

Les solutions au premier de ces deux problèmes reposent sur la conception d'une représentation de l'environnement adéquate pour la navigation, de manière à ce que le robot puisse disposer d'une description de la tâche à accomplir sous forme d'un ensemble d'objectifs à atteindre consécutivement, spécifiés dans l'espace d'observation de son système de perception extéroceptif. Le deuxième problème est bien souvent contourné. Par exemple, en dotant le capteur extéroceptif embarqué de degrés de liberté par rapport à une base mobile non-holonome, la cinématique non-holonome de celle-ci est intégrée dans une modélisation conférant au capteur une cinématique holonome. La tâche de contrôle des déplacements du robot consistant à réguler les informations capteurs sur une référence, la non-holonomie de la base mobile n'affecte alors pas la tâche. En revanche, la pose de la base mobile n'est pas explicitement contrôlée dans cette approche. Il est à noter que cette solution se traduit souvent par une augmentation du poids du capteur ainsi que de son encombrement, ce qui peut être problématique dans certaines applications (pour les robots aériens par exemple).

L'approche de navigation que nous suggérons a pour objectif de proposer une solution viable en réponse aux problématiques énumérées précédemment. Elle repose sur une représentation topologique originale de l'environnement appelée *mémoire sensorielle*. Cette carte contient les informations extéroceptives issues des capteurs embarqués sur le robot mobile et acquises lors d'une phase de mémorisation téléopérée. La mémoire

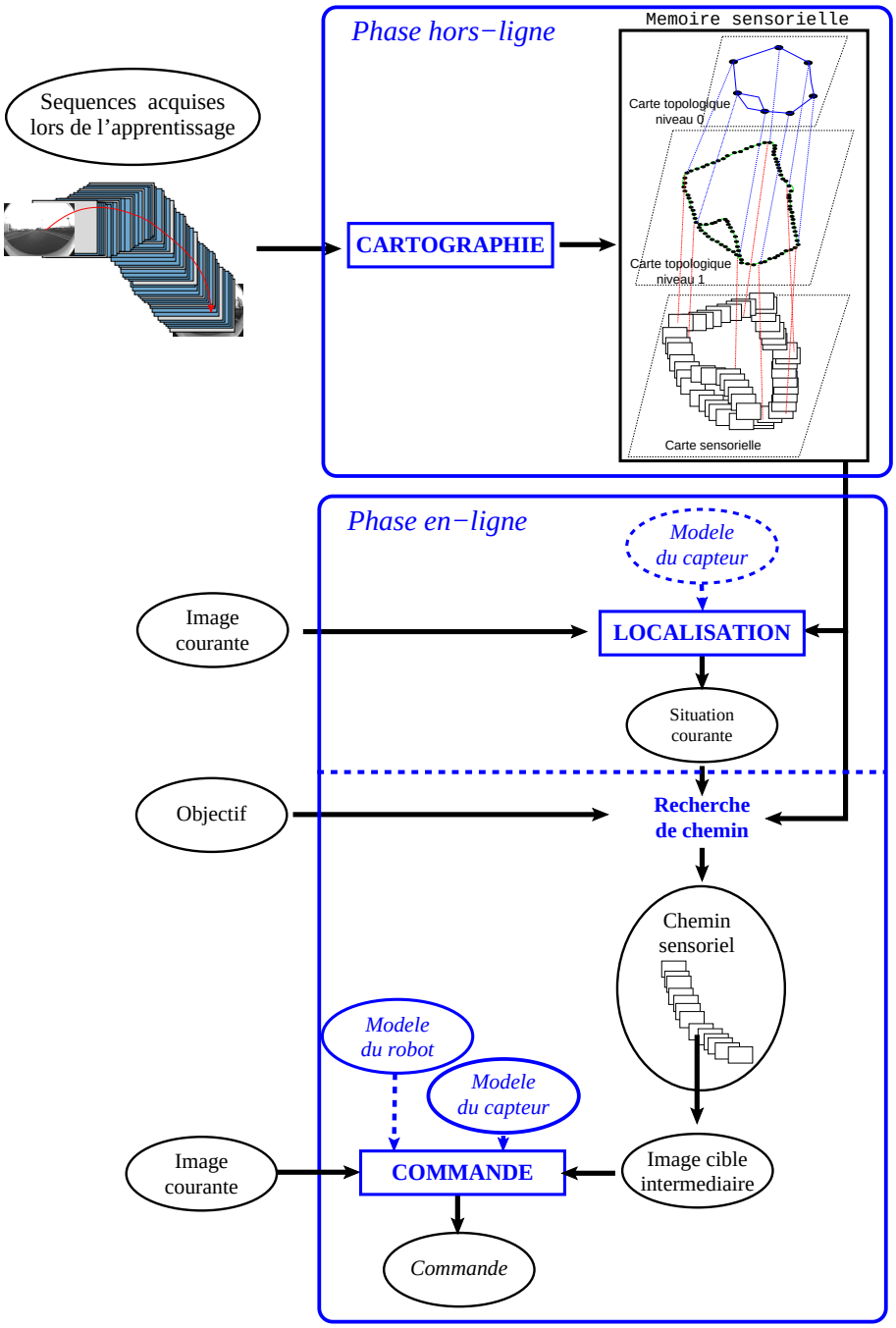


FIG. 1.13 – Notre approche de navigation par mémoire sensorielle.

sensorielle permet alors de définir une stratégie globale de navigation basée sur les informations des capteurs. Cette stratégie, représentée Fig. 1.13, peut être divisée en trois étapes :

1) Construction de la mémoire sensorielle (hors ligne) : lors d'un déplacement téléopéré, une séquence d'images est acquise par le capteur extéroceptif embarqué sur le robot. Afin de réduire la complexité de cette séquence, des images clés sont sélectionnées, organisées et ajoutées à la carte. L'ensemble de ces images forme alors la mémoire sensorielle du robot.

2) Localisation initiale (en ligne) : avant le déplacement autonome, le système robotique est localisé. Cette étape consiste à déterminer l'image de la mémoire sensorielle la plus ressemblante à l'image courante acquise par le capteur extéroceptif.

3) Navigation autonome (en ligne, en temps réel) : étant donnée une image cible contenue dans la mémoire sensorielle, la mission de navigation est définie comme une succession d'images intermédiaires appelée chemin sensoriel, amenant le robot de sa situation courante à la situation cible. Le robot est ensuite commandé le long de ce chemin en utilisant les concepts de la commande référencée capteur.

Tout comme la représentation spatiale de l'environnement s'intègre dans un schéma global de traitement des stimuli et de mémorisation chez les êtres humains, la mémoire sensorielle doit s'insérer dans un schéma de traitement des informations capteurs. Ce schéma peut, par exemple, s'inspirer du modèle de la mémoire décrit Section 1.2.1.

1.3.3 Organisation du manuscrit

Outre cette introduction, ce mémoire de thèse comporte cinq chapitres qui portent respectivement sur les points suivants :

- Le Chapitre 2 a pour objectif de présenter le concept de mémoire sensorielle. Nous exposons tout d'abord la représentation de l'environnement (appelée mémoire sensorielle) exploitée dans notre stratégie de navigation. Le système global de navigation basé sur ce modèle et inspiré des approches cognitives est ensuite décrit.
- Dans le Chapitre 3, nous présentons des stratégies de commande pour le suivi d'un chemin sensoriel pour deux types de robots : les robots mobiles à roues non-holonomes de type char ou bicyclette et les robots aériens de type quadrirotor. Pour chacun d'entre eux, nous précisons tout d'abord l'objectif de commande puis nous proposons des schémas de commande permettant de les atteindre. Des simulations permettent finalement de valider les approches proposées ainsi que d'analyser leurs limites.
- Dans le Chapitre 4, nous étudions le cas des capteurs visuels grand-angle. Nous décrivons alors les éléments mis en place pour chaque étape de notre stratégie de navigation avec ce type de capteur.
- Le Chapitre 5 est dédié à la mise en œuvre complète de notre système de navigation et à sa validation expérimentale. À cette fin, trois robots seront utilisés : un robot

(a) (b) (c)

FIG. 1.14 – Trois types de robots utilisés lors de nos expérimentations : (a) un robot Pioneer, (b) un véhicule urbain électrique de type RobuCab et (c) un drone quadrirotor.

Pioneer 3-AT, un RobuCab et un X4-flyer (voir Fig. 1.14). Nous analysons les résultats obtenus dans différents contextes.

– Finalement, le sixième chapitre dresse un bilan de nos contributions et présente les perspectives qui en découlent.

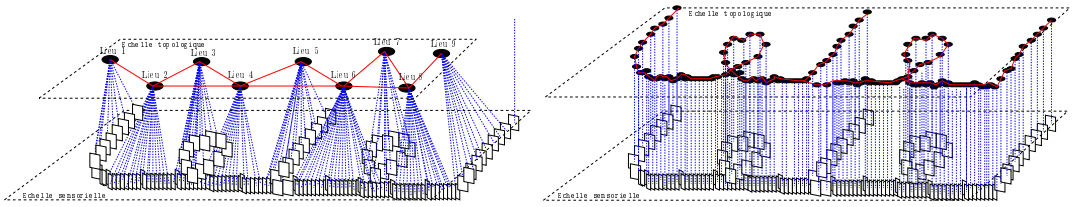
Chapitre 2

Navigation à l'aide d'une mémoire sensorielle

La navigation consiste à déplacer le robot de manière sûre vers une destination donnée. Nous supposons que le robot dispose pour cela d'une représentation de l'environnement. Les besoins auxquels doit répondre cette représentation sont divers comme nous l'avons évoqué dans l'introduction. Tout d'abord, la carte doit permettre la spécification de l'objectif de navigation du robot. Elle doit également supporter sa localisation et la planification de ses déplacements. En outre, afin que ses déplacements soient réalisés de manière sûre, la carte peut intégrer les notions de navigabilité (le robot peut se déplacer sans entrer en collision avec des obstacles) mais également de commandabilité (il existe une commande, appliquée sur un intervalle de temps fini, qui permet de rejoindre la situation cible en partant de la situation initiale). Le robot étant sujet à de grands déplacements, cette représentation peut fournir une description de la tâche à accomplir sous forme d'un ensemble d'objectifs à atteindre consécutivement. Enfin, pour améliorer les possibilités d'adaptation et d'optimisation face aux changements de l'environnement, la carte doit contenir des réseaux de routes.

La représentation par carte sensorielle permet de répondre à ces besoins. En effet, une telle carte est directement définie dans l'espace d'observation du système de perception. De ce fait, ce type de représentation est aisément interprétable et permet de désigner un objectif directement dans cet espace. De plus, les déplacements du robot peuvent être directement générés en utilisant les concepts de la commande référencée capteurs qui est réputée plus robuste vis-à-vis des bruits de mesure et des erreurs de modélisation que les commandes classiques. Dans le but de répondre aux autres besoins que nous avons définis, il est nécessaire d'associer à cette représentation d'autres cartes.

Comme nous l'avons évoqué dans l'introduction, plusieurs solutions sont possibles pour rejoindre un objectif lointain. La première solution consiste à exploiter conjointement les cartes sensorielle et commande. Une image sensorielle est alors associée explicitement à une action comme proposé dans [Matsumoto 96] ou implicitement via



(a) (b)

FIG. 2.1 – Cartes sensorielle et topologique : représentation (a) par lieu, (b) par image.

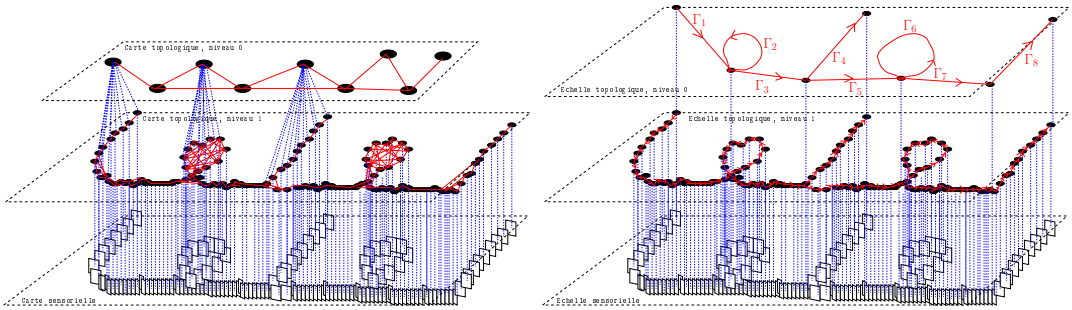
un réseau de neurones comme dans [Gaussier 97, Giovannangeli 06]. Une seconde solution consiste à associer une carte topologique à la carte sensorielle. Un noeud de la carte topologique peut alors représenter un lieu (c'est-à-dire un ensemble d'images, comme dans [Ulrich 00]) (Fig. 2.1 (a)) ou une image unique (Fig. 2.1 (b)). Une arête de la carte topologique définit généralement la navigabilité entre deux noeuds. Dans [Valgren 06, Zivkovic 06, Briggs 06], les arêtes relient les images contenant suffisamment d'amers communs. Dans [Booij 07, Remazeilles 04], le coût entre deux noeuds est défini comme la possibilité de réaliser la tâche de navigation entre ces noeuds. Dans [Booij 07], il dépend de la possibilité de reconstruire la géométrie locale entre les deux noeuds de façon robuste tandis que dans [Remazeilles 04], il est défini de façon empirique à partir de considérations sur les déplacements du robot. Dans [Gaspar 00, Argyros 05, Blanc 05], la définition de l'arête prend en compte la navigabilité et la commandabilité entre deux noeuds. Dans [Santos-Victor 99, Vassallo 00], les arêtes de la carte topologique contiennent également une notion comportementale (suivi de mur, centrage au milieu du couloir, suivi de la séquence d'images ...).

Pour obtenir une représentation de plus haut niveau, plusieurs approches ont été proposées dans la littérature. Une première approche consiste à regrouper les noeuds par lieux (voir Fig. 2.2 (a)) manuellement comme dans [Zhou 03] ou automatiquement comme dans [Zivkovic 06, Booij 07]. Les noeuds peuvent également être regroupés en chemins simples (voir Fig. 2.2 (b)) comme proposé dans [Gaspar 00, Blanc 05] . Ces chemins peuvent, par exemple, représenter une route (en environnement extérieur) ou un couloir (en environnement d'intérieur).

Nous détaillons la représentation que nous avons adoptée dans la Section 2.1. Dans la Section 2.2, nous décrivons le système complet de navigation, inspiré de l'organisation de la mémoire chez l'homme, dans lequel s'intègre la mémoire sensorielle.

2.1 Représentation globale de l'environnement par une mémoire sensorielle

Afin de répondre aux besoins énoncés précédemment, nous proposons de représenter l'environnement par un modèle appelé *mémoire sensorielle* (MS) composé de trois



(a) (b)

FIG. 2.2 – Regroupement des éléments de la carte topologique (a) par lieux, (b) en chemins simples.

cartes ($MS = \{CS, CT1, CT0\}$) : une carte sensorielle (CS) et une carte topologique à deux niveaux ($CT1$ et $CT0$). Ces trois cartes CS, $CT1$ et $CT0$ sont formalisées en utilisant la théorie des graphes et seront décrites en détail dans la suite de cette section.

2.1.1 Quelques définitions

Nous rappelons tout d'abord quelques définitions de base de la théorie des graphes permettant de formaliser la notion de mémoire sensorielle.

Un *graphe* $G = (\mathsf{N}, \mathsf{A})$ est un ensemble fini non vide de nœuds (ou sommets) $\mathsf{N} = \{N_1, N_2, \ldots, N_{n_n}\}$ associé à un ensemble (possiblement vide) d'arêtes (ou arcs) $\mathsf{A} = \{a_1, a_2, \ldots, a_{n_a}\}$ où chaque arête peut être définie par deux sommets $a_r = a_{s,t} = \{N_s, N_t\}$.

La *taille* d'un graphe est le nombre n_a de ses arêtes.

G est un *graphe orienté* ou *digraphe* si tous les éléments de A sont des arêtes orientées ($a_{s,t} \neq a_{t,s}$).

Un *pseudo-graphe* est un graphe dans lequel au moins deux nœuds sont liés par au moins deux arêtes. Un *graphe simple* est un graphe sans boucle dans lequel deux nœuds sont liés par au plus une arête.

Une *chaîne* Γ d'un graphe G est une suite de sommets adjacents. Un *chemin* d'un graphe orienté G est une suite ordonnée de sommets reliés par des arêtes (identique à une chaîne dans un graphe orienté).

Un *graphe connexe* est un graphe dans lequel toute paire de sommets distincts est reliée par une chaîne.

Un *graphe pondéré* est un graphe G suppléé d'une fonction qui associe un poids aux arêtes ($\gamma : \mathsf{A} \rightarrow \Re^+$) ou aux noeuds ($\gamma : \mathsf{N} \rightarrow \Re^+$).

2.1.2 Définitions des cartes

La carte sensorielle est définie par le graphe $CS = (\mathsf{N_{CS}}, \mathsf{A_{CS}})$ avec :

$$\begin{cases} \mathsf{A_{CS}} \text{ un ensemble vide,} \\ \mathsf{N_{CS}} = \left\{ \mathcal{I}_j^i \mid j \in \{1, 2, \ldots, n_{\Gamma^i}\} \mid i \in \{1, 2, \ldots, n_\Gamma\} \right\} \end{cases}$$

où $\mathcal{I}_j^i$ est la j$^{\text{ème}}$ image clé du i$^{\text{ème}}$ chemin, n_Γ représente le nombre de chemins simples considérés dans l'environnement et n_{Γ^i} est le nombre d'images du i$^{\text{ème}}$ chemin.

Le premier niveau de la carte topologique est défini par le graphe : $CT1 = (\mathsf{N_{CT1}}, \mathsf{A_{CT1}})$. Chaque noeud de $CT1$ correspond à un lieu de l'environnement défini par une image unique de la carte sensorielle. Les noeuds de la carte $CT1$ sont alors définis comme suit :

$$\mathsf{N_{CT1}} = \left\{ \left\{ N_j^i \mid j \in \{1, 2, \ldots, n_{\Gamma^i}\} \right\} \mid i \in \{1, 2, \ldots, n_\Gamma\} \right\}$$

où le noeud N_j^i correspond au lieu d'acquisition de l'image $\mathcal{I}_j^i$ et où les noeuds N_j^i et N_{j+1}^i sont liés par une arête. Les propriétés des arêtes $\mathsf{A_{CT1}}$ de ce graphe seront détaillées dans la suite. Pour des raisons pratiques, il est nécessaire que ce graphe $CT1$ soit connexe. Cela garantit qu'il est possible d'atteindre n'importe quel noeud du graphe depuis n'importe quel autre noeud.

Les chemins simples de l'environnement sont représentés dans la carte topologique $CT0$ définie par le graphe $CT0 = (\mathsf{N_{CT0}}, \mathsf{C_{CT0}})$. Dans la suite, nous nommerons ces chemins des *séquences* afin de les différencier des chemins tels que définis dans la théorie des graphes. Les n_Γ séquences de $CT0$ sont des chemins simples obtenus par concaténation des arêtes de $CT1$ joignant deux noeuds de $CT0$. Ce regroupement a plusieurs avantages. Tout d'abord, une séquence peut représenter un couloir ou une route, lieu le long duquel se déplace le robot tandis qu'un carrefour entre deux séquences correspond à un lieu où le robot peut bifurquer. De plus, le robot mémorisant des séquences d'images lors de la phase d'apprentissage, la phase de construction de la carte est simplifiée. Les noeuds de la carte sont alors définis comme suit :

$$\mathsf{N_{CT0}} = \{\Gamma^1, \Gamma^2, \ldots, \Gamma^{n_\Gamma}\}$$

où Γ^i représente la i$^{\text{ème}}$ séquence :

$$\Gamma^i = \left\{ N_j^i \mid j \in \{1, 2, \ldots, n_{\Gamma^i}\} \right\}$$

Une arête $c = c_{i,j}$ appartenant à $\mathsf{C_{CT0}}$ relie le noeud final $N_{n_{\Gamma^i}}^i$ de la séquence Γ^i et le noeud initial N_1^j de la séquence Γ^j. Elle est définie de la même manière que les arêtes du graphe $CT1$ (voir Section 2.1.3).

Afin de définir les critères d'optimalité exploités lors de la phase de planification de chemin, il est possible d'associer une pondération aux cartes topologiques. Le graphe $CT1$ est alors muni d'une fonction de pondération γ :

$$\left\{ \begin{array}{ccc} \mathsf{A_{CT1}} & \mapsto & \Re^+ \\ a_{i,j} & \rightarrow & \gamma(a_{i,j}) \end{array} \right. \tag{2.1}$$

Plusieurs choix de pondération sont possibles. Les notions de navigabilité et de commandabilité étant implicitement contenues dans $\mathsf{A_{CT1}}$ (comme nous le verrons dans le paragraphe suivant), il n'est pas nécessaire de les introduire via la fonction de pondération. On peut alors envisager une pondération unitaire ou des fonctions de pondération :

- dépendant des données capteurs (par exemple de la qualité des appariements entre les images $\mathcal{I}_i$ et $\mathcal{I}_j$),
- dépendant de la commande (par exemple de la qualité de l'estimation des entrées de commande à partir des données contenues dans N_i et N_j).

La fonction de pondération de $CT0$ est, quant à elle, obtenue directement à partir de celle de $CT1$:

$$\left\{ \begin{array}{ccc} \mathsf{N_{CT0}} & \mapsto & \Re^+ \\ \Gamma^i & \rightarrow & \gamma(\Gamma^i) = \sum_{j=1}^{n_{\Gamma^i}-1} \gamma(a_{N_j^i, N_{j+1}^i}) \end{array} \right. \tag{2.2}$$

Un exemple de mémoire sensorielle est représenté Figure 2.3 dans le cadre de la navigation d'un véhicule urbain à roues muni d'une caméra grand-angle. Sur cette figure, uniquement les images correspondant aux noeuds extrêmes des séquences de $CT1$ sont représentées. Cette représentation contient à la fois des observations de l'environnement facilement interprétables par l'utilisateur (les images) et un modèle de haut niveau permettant de comprendre le réseau des routes.

2.1.3 Propriétés des arêtes des cartes topologiques

Comme nous l'avons évoqué au début de ce chapitre, la navigabilité entre les noeuds est une information qui nous parait indispensable dans une stratégie de navigation. Cette notion est directement intégrée aux arêtes du graphe $CT1$ construites sous l'hypothèse suivante :

Hypothèse 1 *Soient les repères de commande $\mathcal{F}_i$ et $\mathcal{F}_j$ associés respectivement au robot pour les noeuds N_i et N_j liés par une arête orientée $a = (N_i, N_j)$. Alors, il existe un chemin admissible (Υ) allant de $\mathcal{F}_i$ à $\mathcal{F}_j$ pour le robot dont on connaît les contraintes de déplacement.*

La représentation de l'environnement MS permet la description de la tâche de navigation sous la forme d'une séquence d'objectifs intermédiaires. La commande doit alors être estimée à partir des informations capteurs courantes et de consigne tout au long

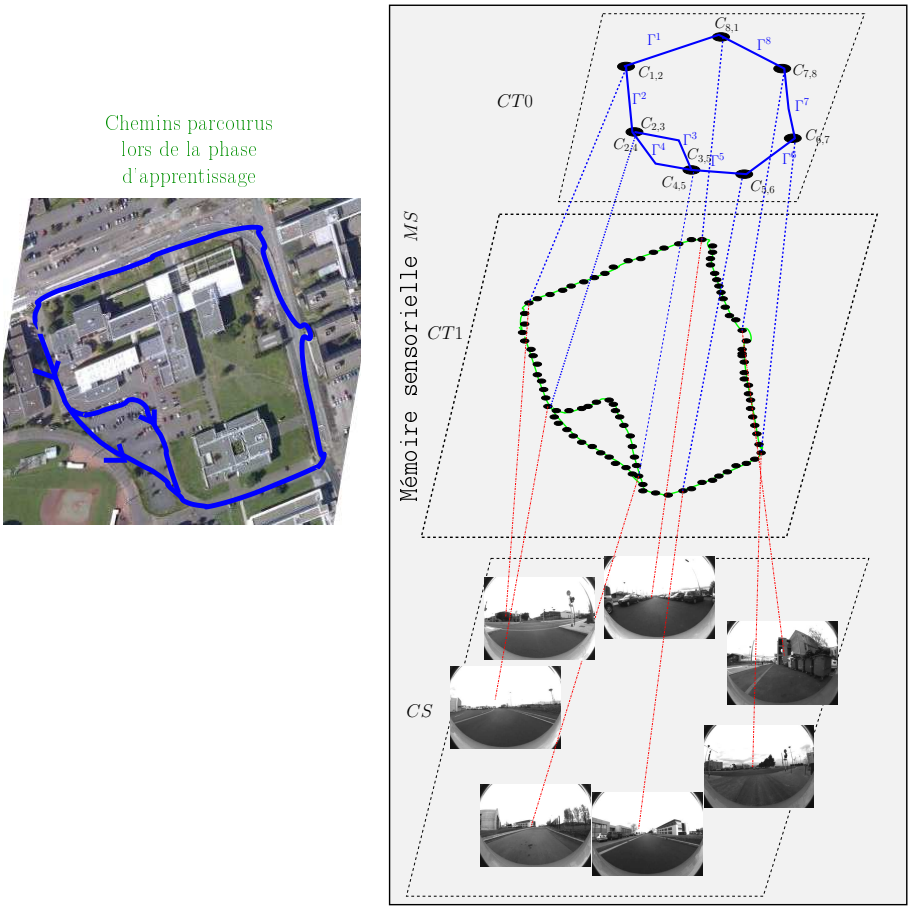


FIG. 2.3 – Mémoire visuelle d'un robot urbain à roues obtenue avec une caméra grand angle.

du chemin à parcourir. De manière implicite, nous faisons donc la seconde hypothèse suivante :

Hypothèse 2 *Les images $\mathcal{I}_i$ et $\mathcal{I}_j$ correspondant aux noeuds N_i et N_j, liés par une arête orientée $a = (N_i, N_j)$, sont telles qu'il est possible d'estimer les entrées de commande tout au long du chemin (Υ) parcouru entre $\mathcal{F}_i$ et $\mathcal{F}_j$.*

On note que l'hypothèse 2 assure la commandabilité du robot entre deux situations successives. Elle ne signifie pas que l'état complet du robot doit être à estimer dans un repère absolu comme nous le verrons plus précisément dans les chapitres 3 et 4.

2.2 Système complet de navigation à l'aide d'une mémoire

Nous venons de définir l'organisation de la mémoire sensorielle. Cette mémoire peut s'insérer, comme évoqué dans le contexte de la psychologie cognitive, dans un schéma global de navigation allant du traitement du stimulus jusqu'au déplacement autonome. Nous proposons un système global de navigation inspiré de l'organisation de la mémoire chez l'homme présentée dans la Section 1.2.2. Ce système peut être décomposé en trois modules : le registre sensoriel, la mémoire de travail et la mémoire à long terme (voir Figure 2.4). Le *registre sensoriel* filtre les données acquises par les capteurs en fonction de la tâche à effectuer. La mémoire à long terme contient la représentation spatiale de l'environnement, ici, la mémoire sensorielle (MS) ainsi que les informations complémentaires utiles à la réalisation de la tâche de navigation (modèle du robot et modèle du capteur par exemple). Enfin, les différents processus impliqués dans la tâche de navigation sont effectués au niveau de la mémoire de travail à partir d'informations provenant à la fois de la mémoire à long terme et du registre sensoriel. Comme nous l'avons évoqué dans l'introduction et comme illustré Fig. 1.13, ces processus sont la construction de la carte que nous décrivons Section 2.2.1, la localisation initiale, détaillée Section 2.2.2 et la commande le long d'un chemin sensoriel, présentée Section 2.2.3.

2.2.1 Construction de la mémoire sensorielle

L'étape de construction de MS se déroule hors ligne. Lors d'une phase d'apprentissage, le robot est téléopéré dans son environnement de travail afin d'acquérir une séquence d'images $S = \{\mathcal{I}[i] \,|\, i \in \{1, 2, \ldots, n_S\}\}$ (un exemple de trajets suivis est représenté Figure 2.5).

Les trois cartes de la mémoire sensorielle sont alors construites simultanément en trois étapes :

Étape 1 : les images clés $\{\mathcal{I}[k] \,|\, k \subset \{1, 2, \ldots, n_S\}\}$ sont sélectionnées au niveau de la mémoire de travail et transférées vers la mémoire à long terme (voir Figure 2.6) en respectant les hypothèses (1) et (2).

Étape 2 : pour chaque image clé $\mathcal{I}[k]$:
- $\mathcal{I}[k]$ est ajoutée à la carte sensorielle CS,

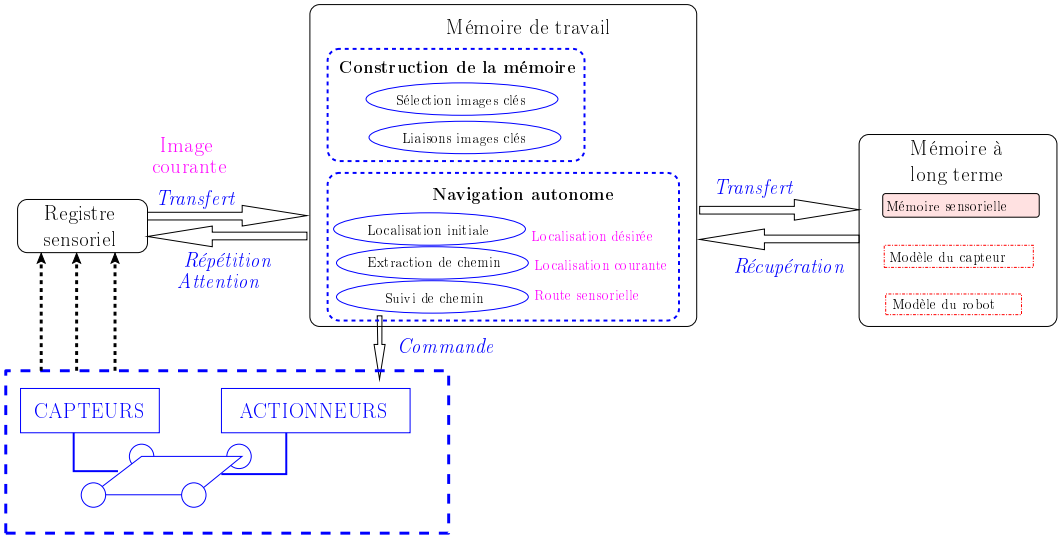


FIG. 2.4 – Système complet de navigation par mémoire sensorielle.

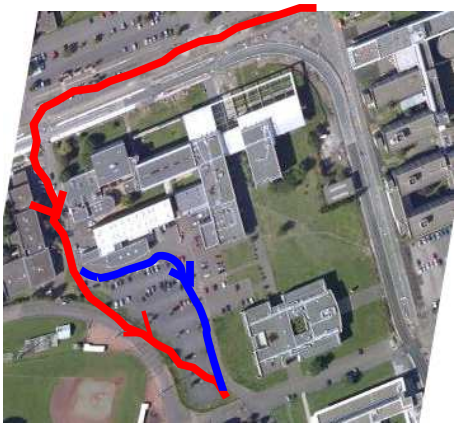


FIG. 2.5 – Exemple de chemins parcourus lors de la phase d'apprentissage.

- un noeud N_k correspondant est ajouté à la carte topologique $CT1$,
- si cette image n'est pas la première ajoutée, une arête liant N_k au noeud N_{k-1} est créée dans $CT1$.

L'ensemble de ces noeuds et arêtes permet de définir une nouvelle séquence dans $CT0$. Cette étape est représentée Figure 2.7.

Étape 3 : des noeuds de séquences différentes sont reliés manuellement par des arêtes (lorsque cela est possible) afin de compléter le graphe $CT0$ (voir Fig. 2.8).

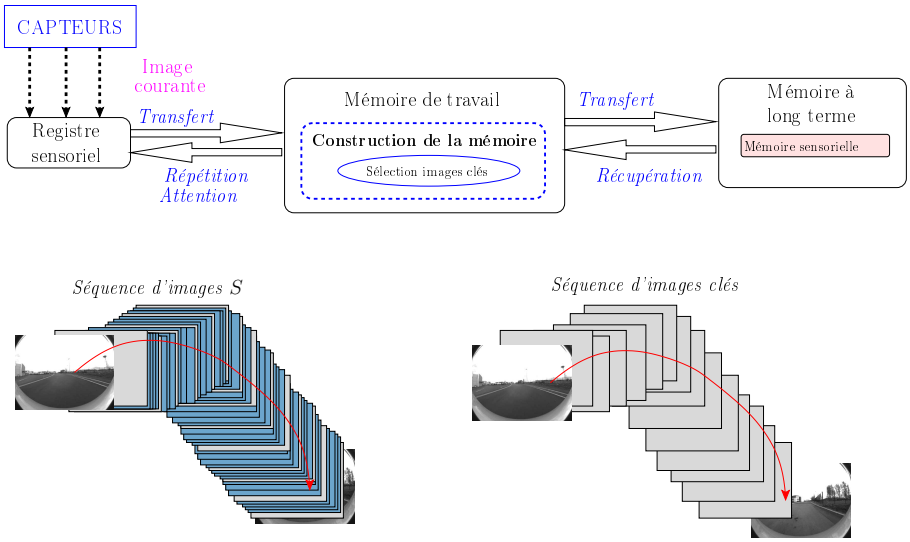
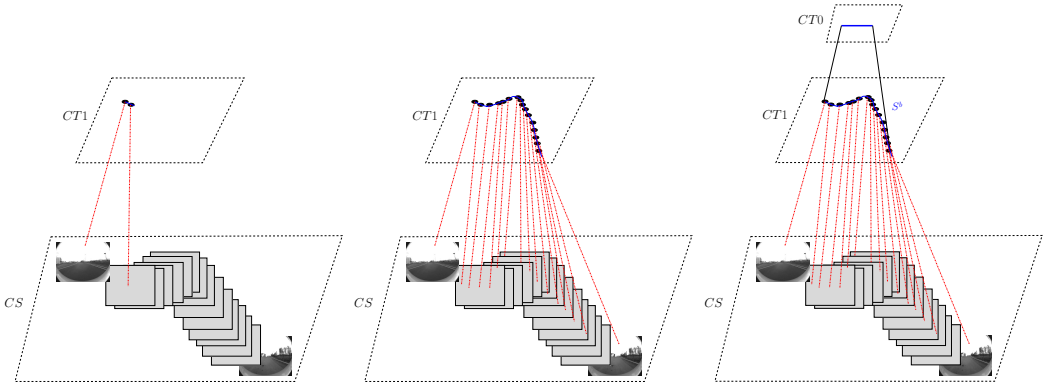


FIG. 2.6 – Étape 1 de la construction de la mémoire : sélection des images clés dans la mémoire de travail.

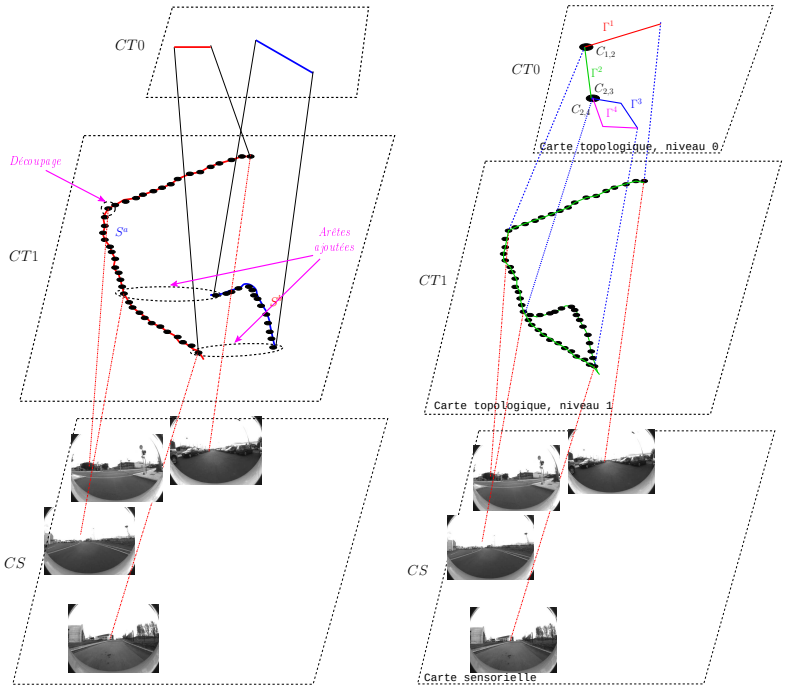


(a) Ajout d'une image clé, d'un noeud et d'une arête

(b) Séquence d'images clés importée

(c) Création de la séquence dans $CT0$

FIG. 2.7 – Étape 2 de la construction de la mémoire : ajouts des images clés dans CS, des noeuds correspondants dans $CT1$ (avec les arêtes) puis de la séquence dans $CT0$.



(a) Ajout d'arêtes et mise à jour de $CT0$

(b) Mémoire sensorielle obtenue

FIG. 2.8 – Étape 3 de la construction de la mémoire : mise à jour de $CT0$.

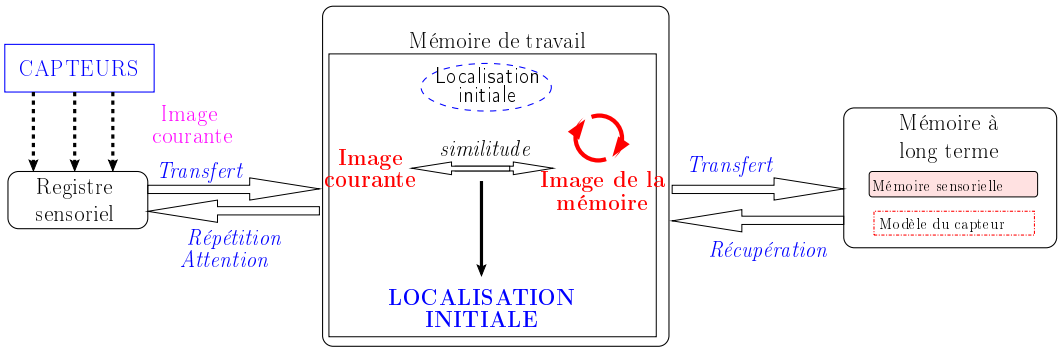


FIG. 2.9 – Système complet de navigation par mémoire sensorielle : localisation initiale.

2.2.2 Localisation initiale

La première tâche à réaliser en ligne lors de la navigation autonome consiste en la localisation initiale du robot dans sa mémoire sensorielle MS. Nous avons choisi d'employer une localisation par inférence directe car celle-ci permet la localisation sans déplacement du robot (voir Section 1.1.3). Cette tâche peut être conduite en comparant l'image courante $\mathcal{I}_c$ aux images de la carte sensorielle CS (voir Figure 2.9). Cela suppose qu'il existe une fonction d permettant de mesurer la distance (ou *similarité*) entre deux images. La localisation est alors l'image $\mathcal{I}_{i_0}^{j_0}$ de CS telle que :

$$\mathcal{I}_{i_0}^{j_0} = arg \min_{\mathcal{I}_i^j \in \mathsf{N}_{CS}} d(\mathcal{I}_c, \mathcal{I}_i^j)$$

Afin de réduire le coût calculatoire de cette étape, la dimension de l'espace de recherche peut être diminuée en exploitant des données fournies par des capteurs complémentaires ou bien par l'utilisateur. On a alors :

$$\mathcal{I}_{i_0}^{j_0} = arg \min_{\mathcal{I}_i^j \in \mathsf{N}'_{CS}} d(\mathcal{I}_c, \mathcal{I}_i^j)$$

où $\mathsf{N}'_{CS} \subset \mathsf{N}$ est un sous-ensemble d'images de CS. N'_{CS} peut, par exemple, être obtenu connaissant la séquence Γ^i de $CT0$ sur laquelle est situé initialement le robot : $\mathsf{N}'_{CS} = \{\mathcal{I}_j^i \,|\, j \in \{1, 2, \ldots, n_{\Gamma^i}\}\}$.

Nous détaillerons cette étape de localisation initiale lorsque les images sont acquises par une caméra grand-angle dans la Section 4.2.

2.2.3 Navigation autonome

On suppose maintenant que la localisation initiale $\mathcal{I}_1 = \mathcal{I}_{i_0}^{j_0}$ et l'objectif de navigation $\mathcal{I}_n = \mathcal{I}_{i_f}^{j_f}$ sont connus dans CS. La navigation autonome de robot est alors conduite en deux étapes. La première étape consiste à extraire de MS un chemin sensoriel Ψ liant $\mathcal{I}_1$ à $\mathcal{I}_n$:

$$\Psi = \{\mathcal{I}_i \,|\, i \in \{1, 2, \ldots, n\}\}$$

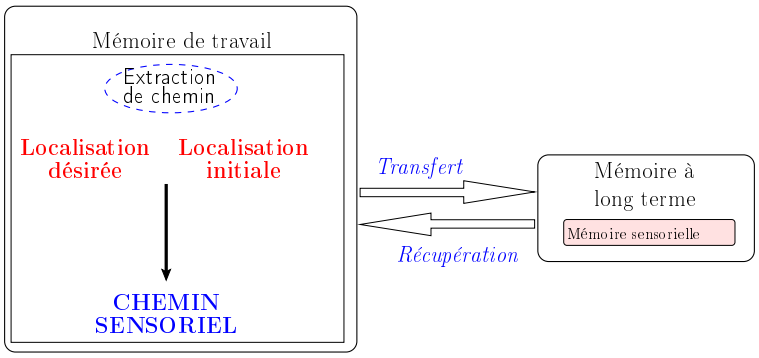


FIG. 2.10 – Système complet de navigation par mémoire sensorielle : planification de chemin

La seconde étape consiste alors au suivi de Ψ.

Extraction du chemin sensoriel L'extraction de Ψ emploie les cartes topologiques $CT1$ ou $CT0$. Cette étape, effectuée en ligne, est représentée Fig. 2.10. De manière synthétique, connaissant $\mathcal{I}_{i_0}^{j_0}$ et $\mathcal{I}_{i_f}^{j_f}$, il est possible et aisé de retrouver les nœuds correspondants $N_{i_0}^{j_0}$ et $N_{i_f}^{j_f}$ dans $CT1$ ainsi que les chaînes Γ^{j_0} et Γ^{j_f} correspondantes dans $CT0$. La construction de Ψ peut alors être conduite via des algorithmes de recherche du plus court chemin bien connus en théorie des graphes tels que les algorithmes de Dijkstra ou de Bellman–Ford (voir, par exemple, [Cherkassky 96]).

Afin de réduire le coût calculatoire de cette étape, il est préférable d'utiliser la carte $CT0$ plutôt que la carte $CT1$ car ce coût croît avec la taille du graphe de recherche. On note $\Gamma^{j_1}, \Gamma^{j_2} \ldots \Gamma^{j_{f-1}}$ la succession de séquences du graphe $CT0$ permettant de lier Γ^{j_0} à Γ^{j_f} (via le plus court chemin). Alors les images du chemin sensoriel définissant Ψ correspondent aux noeuds obtenus par concaténation des différents séquences $\Gamma^{j'_0}$ à $\Gamma^{j'_f}$:

$$\Gamma = \Gamma^{j'_0} \oplus \Gamma^{j_1} \oplus \Gamma^{j_2} \oplus \ldots \Gamma^{j_{f-1}} \oplus \Gamma^{j'_f}$$

avec :

$$\begin{cases} \Gamma^{j'_0} = \{N_i^{j_0} \,|\, i \in \{i_0,\, i_0+1,\, i_0+2, \ldots, n_{\Gamma^{j_0}}\}\} \\ \Gamma^{j'_f} = \{N_i^{j_f} \,|\, i \in \{1,\, 2, \ldots j_f-2,\, j_f-1,\, j_f\}\} \end{cases}$$

et où l'opérateur de concaténation est défini comme suit :

$$\Gamma^{p_1} \oplus \Gamma^{p_2} = \{N_i^{p_{1|2}} | i \in \{1,\, 2, \ldots, n_{p_1},\, n_{p_1}+1, \ldots n_{p_1}+n_{p_2}\}\}$$

$$\text{avec } N_i^{p_{1|2}} = \begin{cases} N_i^{p_1} & \text{si} \quad i < n_{p_1} \\ N_{i-n_{p_1}}^{p_2} & \text{si} \quad n_{p_1} < i \leq n_{p_1}+n_{p_2} \end{cases}.$$

Un exemple de chemin sensoriel est représenté Fig. 2.11. Dans cet exemple, le chemin sensoriel Ψ joignant l'image initiale $\mathcal{I}_{i_0}^{j_0}$ à l'image cible $\mathcal{I}_{i_f}^{j_f}$ est obtenu à partir du chemin Γ défini dans $CT1$: $\Gamma = \Gamma^{4'} \oplus \Gamma^5 \oplus \Gamma^6 \oplus \Gamma^7$ où $\Gamma^{4'} = \{N_i^4 | i \in \{i_0,\, i_0+1, \ldots, n_{\Gamma^4}\}\}$.

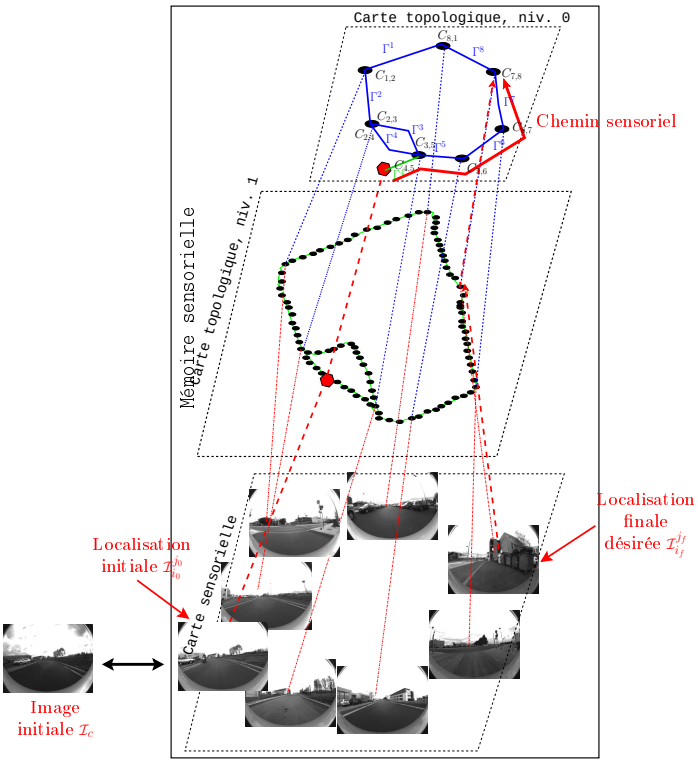


FIG. 2.11 – Extraction du chemin sensoriel Ψ liant l'image initiale $\mathcal{I}_{i_0}^{j_0}$ à l'image cible $\mathcal{I}_{i_f}^{j_f}$.

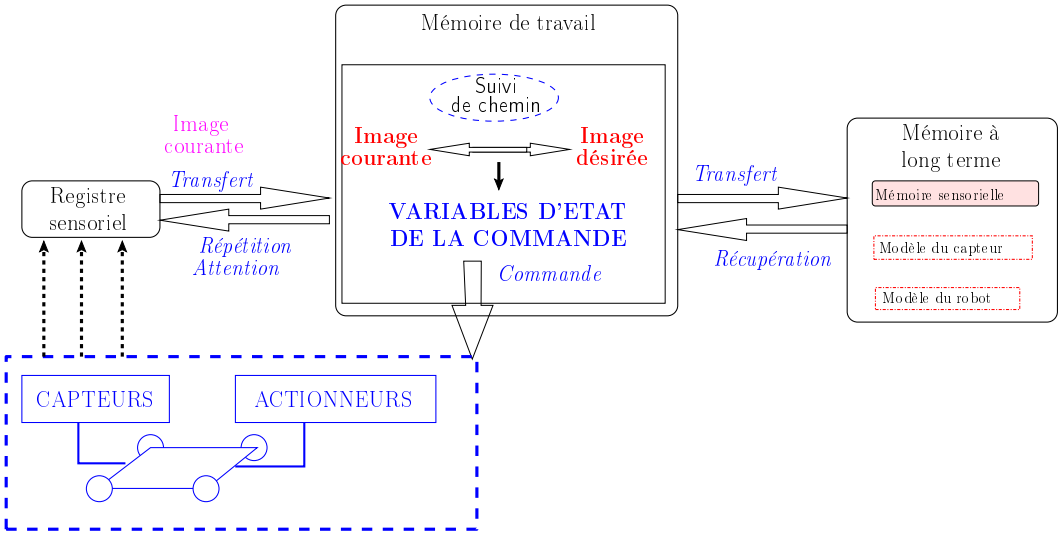


FIG. 2.12 – Système complet de navigation par mémoire sensorielle : suivi de chemin.

Suivi de chemin Comme nous l'avons évoqué précédemment, le chemin sensoriel Ψ est défini par une succession d'images capteurs. Afin d'en assurer le suivi, il est alors possible de s'affranchir d'une localisation géométrique absolue du robot (et ainsi accroître la robustesse vis-à-vis des erreurs de modélisation et des bruits de mesure) en exploitant les concepts de la commande référencée capteurs. Le registre sensoriel fournit les données capteurs courantes tandis que les données capteurs de consigne successives sont extraites de la mémoire sensorielle MS (voir Fig. 2.12). Lorsque l'image désirée $\mathcal{I}_i$ a été atteinte, la localisation du robot est mise à jour (approche de suivi d'hypothèse explicite). La tâche de navigation autonome se termine lorsque l'image finale du chemin sensoriel a été atteinte.

Nous proposerons dans le Chapitre suivant des lois de commande adaptées au suivi d'un chemin sensoriel dans le cas de robots mobiles à roues non-holonomes de type char ou bicyclette ainsi que dans le cas de robots aériens de type quadrirotor.

2.3 Conclusion

Dans ce chapitre, nous avons détaillé la représentation de l'environnement exploitée par notre stratégie de navigation. Nous avons vu que celle-ci permet d'alimenter tous les processus nécessaires à la navigation autonome référencée capteur d'un robot mobile (localisation, planification et commande). De manière assez remarquable, elle intègre en outre implicitement les notions de navigabilité et de commandabilité. Constituée par un réseau d'observations capteurs organisées, elle donne également une vue globale de l'environnement aisément exploitable pour la planification de chemins vers des objectifs lointains via la théorie des graphes.

Lors du développement de la représentation de l'environnement par mémoire sensorielle, deux hypothèses ont été faites. L'hypothèse 1 est indispensable afin d'assurer la navigabilité de l'environnement de déplacement du robot. Afin d'en assurer la validité, il serait souhaitable (même si nous ne l'avons pas traité dans cette thèse) d'intégrer la notion d'espace libre à la représentation par mémoire sensorielle. Cette notion, dont les problématiques de perception sous-jacentes sont très complexes, devrait s'intégrer assez aisément à la mémoire sensorielle, par exemple, via l'ajout des informations pertinentes au niveau des noeuds de $CT1$.

La seconde hypothèse suppose que les entrées de commande puissent être estimées le long de toute arête reliant deux noeuds de $CT1$. Notons tout d'abord que cela ne suppose pas que la pose (position et orientation) du robot soit connue dans un repère absolu. Plusieurs situations répondant à cette hypothèse peuvent être rencontrées :

- le capteur fournit la localisation absolue du robot (et par conséquent celle entre deux noeuds). Dans ce cas, la localisation doit être connue de manière exacte et non bruitée afin de réaliser un déplacement précis sur une distance importante. Ces hypothèses restant difficilement acceptables en pratique, nous ne nous intéresserons pas à cette situation.
- le capteur fournit une localisation précise entre deux noeuds. Cela peut, par exemple, être le cas avec un système stéréoscopique ou encore avec un scanner 3D. Dans ce cas, comme nous le verrons dans le chapitre 3, l'hypothèse 2 est vérifiée.
- le capteur fournit partiellement la localisation entre deux noeuds. C'est notamment le cas d'une caméra embarquée sur le robot, la position étant alors connue à une constante multiplicative près (facteur d'échelle). Cette situation constituera notre cas d'étude. Nous verrons alors qu'il est possible de construire des lois de commande pour les robots mobiles à roues et les drones quadrirotors permettant de suivre un chemin sensoriel afin d'assurer la validité de l'hypothèse 2.

Nous n'avons traité ici que des solutions avec un capteur unique. Cependant, il est également possible de coupler plusieurs capteurs afin de répondre à l'hypothèse 2. Le système perceptuel obtenu doit alors fournir une localisation précise ou partielle entre deux noeuds. Les données provenant de plusieurs capteurs s'intègrent assez aisément à la mémoire sensorielle via leur ajout dans CS et la mise en correspondance des images avec les noeuds de $CT1$.

Chapitre 3

Commande de robots le long d'un chemin sensoriel

La troisième étape de notre stratégie de navigation consiste tout d'abord à extraire de la mémoire un chemin sensoriel Ψ puis ensuite à commander le robot le long de Ψ. Dans ce chapitre, nous nous focalisons sur cette seconde phase (voir Fig. 3.1) pour deux types de robots : les robots mobiles à roues non-holonomes[1] (sujets de nos applications au LASMEA) et les robots aériens de type quadrirotor (sujets d'applications au CEA-LIST). Nous présentons dans la Section 3.1 un bref état de l'art sur la commande de ces robots et sur le suivi de chemin sensoriel. Nous traitons du suivi de chemin sensoriel pour les robots mobiles à roues non-holonomes dans les Sections 3.2 et 3.3 et pour les quadrirotors dans les Sections 3.4 et 3.5.

3.1 Commande des robots mobiles

Pour les robots à roues et volants, différents types de tâches ont été étudiés dans la littérature. Nous présentons ici les principales tâches assignées à ces robots avec un intérêt particulier pour celles de suivi de chemin sensoriel.

3.1.1 Commande des robots mobiles à roues

Trois types de tâches peuvent être distingués pour les robots mobiles à roues : la stabilisation, la poursuite de trajectoire et le suivi de chemin. La stabilisation a pour objectif d'asservir le robot sur une pose (position et orientation) fixe. La résolution de ce problème pour un robot non-holonome est difficile car elle ne peut pas être réalisée au moyen d'une loi de commande par retour d'état continu (théorème de Brockett

[1] Un système est dit non-holonome lorsque les contraintes de déplacement sont sous la forme d'équations différentielles non complètement intégrables c'est-à-dire que certaines directions d'évolution ne sont pas instantanément possibles. Par exemple, pour les robots à roues de ce type, l'évolution latérale n'est pas réalisable de façon instantanée.

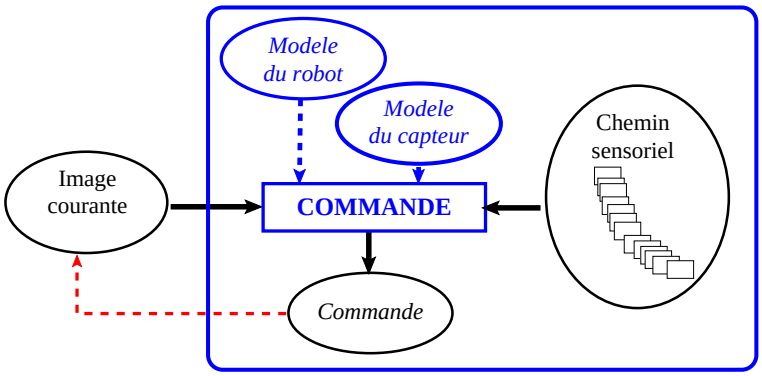


FIG. 3.1 – Suivi du chemin sensoriel.

[Brockett 83]). Pour le second type de tâche, il s'agit de réaliser le suivi d'une trajectoire ne comportant pas de points d'arrêt, selon une loi horaire donnée. Enfin, pour le dernier type de tâche, la trajectoire à suivre n'est pas associée à une loi horaire. Pour résoudre les problèmes de suivi et de poursuite dans le cas non-holonome, plusieurs pistes ont été explorées dans la littérature. La première piste consiste à construire des lois de commande non linéaires via des fonctions de Lyapunov comme dans [Zhang 02]. La principale difficulté est alors de définir une fonction de Lyapunov satisfaisante. Une seconde solution consiste à linéariser exactement les modèles non linéaires associés aux robots non-holonomes afin d'exploiter les concepts de l'automatique linéaire. Cependant, les modèles cinématiques ne sont généralement pas complètement linéarisables au moyen de retours d'état statique. En partant du constat que dans de nombreuses applications, la plage de fonctionnement des véhicules reste dans le voisinage d'une configuration donnée, la synthèse des lois de commande peut être simplifiée en linéarisant le modèle du véhicule autour de celle-ci comme dans [Broggi 99]. Toutefois, le modèle linéarisé n'est pas valable en dehors du voisinage de cette configuration, ce qui se traduit par des difficultés lorsque le chemin présente des variations géométriques importantes. Afin de commander le robot en tenant compte de sa non-holonomie et sur toute sa plage de fonctionnement, des retours d'état discontinu [Astolfi 96] ou dynamique [De Luca 00] ont été proposés. Une autre approche possible consiste à convertir les modèles non-linéaires sous une forme dite chaînée. Il est ensuite possible d'obtenir une expression linéaire de l'écart par rapport à la trajectoire de référence [Samson 95]. Nous reviendrons plus en détails sur cette méthodologie que nous avons adoptée dans la Section 3.2.3.

Le problème de suivi de chemin sensoriel, qui nous intéresse plus particulièrement dans le cadre de nos travaux, a également été abordé sous divers angles dans la littérature. Il peut tout d'abord être vu comme un problème de stabilisation en une succession de poses correspondant aux prises de vue des images du chemin sensoriel de référence. Pour les robots holonomes, ce problème est largement simplifié et peut, par exemple, être traité via les stratégies de retour au nid (*homing* en anglais). La thèse de S. Gourichon [Gourichon 04] donne un état de l'art complet de cette stratégie introduite

par les biologistes Cartwright et Collett dans [Cartwright 83]. Celle-ci, inspirée par les systèmes de navigation des abeilles et des fourmis, consiste à déduire un vecteur de mouvement pointant vers la situation cible à partir des observations capteurs courantes et désirées. Dans sa forme de base, le retour au nid est une stratégie locale. Cette approche a été étendue au suivi d'une séquence d'images dans [Argyros 01, Argyros 05] puis a été reprise dans [Goedemé 05, Kirigin 05]. Les stratégies de retour au nid ne sont, cependant, pas adaptées au cas des robots mobiles à roues non-holonomes car les stabilisations successives ne peuvent pas être réalisées au moyen d'une loi de commande par retour d'état continu avec un capteur rigidement lié au robot. Cette difficulté peut, cependant, être contournée en dotant le capteur de degrés de liberté supplémentaires afin de lui conférer une cinématique holonome. Cette méthodologie a été introduite dans [Pissard-Gibollet 91, Pissard-Gibollet 95] avec une caméra montée sur un bras manipulateur puis reprise dans [Cadenat 99, Folio 07]. L'objectif de commande est alors formulé comme une tâche de positionnement de la caméra et est réalisé par asservissement visuel.

Le parcours automatique d'un chemin sensoriel peut également être traité comme une tâche de suivi de chemin. Pour cela, on peut se contenter de commander uniquement la vitesse angulaire du robot et ainsi laisser libre la vitesse longitudinale„ par exemple, fixée par l'utilisateur. Plusieurs stratégies ont été proposées dans la littérature en utilisant des informations extraites d'images visuelles. Dans [Rivlin 03, Šegvić 07, Diosi 07], la vitesse angulaire est choisie proportionnelle à une erreur dépendant des positions des points dans les images courante et désirée. Dans [Cherubini 09], plusieurs approches d'asservissement visuel sont proposées. Les lois de commande décrites permettent une décroissance de l'erreur dans l'image de façon exponentielle mais le comportement 3D du robot n'est pas maîtrisé. Dans [Becerra 08], la vitesse angulaire permet de réguler la position de l'épipole par rapport à l'image désirée. Une loi de commande par modes glissants est employée dans ce but. Dans [Blanc 05], le chemin de référence pour atteindre l'image intermédiaire à rejoindre est défini comme la droite passant par le centre de commande et dans la direction du robot lors de la prise de vue. Une loi de commande basée sur la théorie des systèmes chaînés permet alors de suivre ce chemin.

En ce qui nous concerne, nous n'avons pas souhaité équiper le capteur d'un système mécanique complémentaire. Nous nous focalisons donc sur des approches adéquates pour un capteur fixé sur la base du robot mobile et non-actionné. Les contraintes de mouvement de la caméra sont alors identiques à celles du robot. Afin d'assurer le meilleur confort pour les passagers ainsi que pour le matériel, nous souhaitons que le robot se déplace avec une vitesse longitudinale donnée (constante ou variable). En outre, il est important de maîtriser le plus possible la trajectoire 3D parcourue afin d'assurer un déplacement sûr du véhicule. Nous verrons dans la suite comment atteindre ces différents objectifs (se référer à la Section 3.2). Dans la Section 3.3, nous étudierons en simulations les performances de l'approche proposée.

3.1.2 Commande des quadrirotors

Le suivi d'un chemin sensoriel par les robots aériens autonomes de petites dimensions comme le quadrirotor du CEA a fait l'objet de peu de travaux. En effet, les recherches sur ce type d'engins n'ont pris un réel essor que depuis une dizaine d'années. Des travaux ont porté sur la conception de l'engin (structure mécanique, énergie, électronique embarquée, capteurs ...) comme dans [Pounds 02, Bouabdallah 04b]. L'estimation de l'état du drone (orientation, position, vitesse) est également un axe de recherche important. L'orientation peut être estimée en filtrant les données fournies par la centrale inertielle [Hamel 06b] ou à partir de la position de l'horizon dans l'image [Demonceaux 06]. L'orientation et la position d'un drone peuvent également être obtenus par une fusion des informations provenant des différents capteurs embarqués (GPS, centrale inertielle, caméra) comme dans [Wu 05].

Plusieurs types de tâches peuvent être distingués pour la commande de ces engins. La première tâche est la stabilisation en assiette du drone (problème non linéaire). Dans [Bouabdallah 04a, Lozano 05] par exemple, le modèle de l'engin est linéarisé à l'équilibre et une commande PID (Proportionnelle, Intégrale, Dérivée), dissociée sur chaque axe de rotation, permet de stabiliser l'assiette. Des commandes non-linéaires, basées sur les angles d'Euler ont également été proposées et permettent la convergence exponentielle de l'orientation (voir [Benallegue 07] par exemple). Toutefois, l'utilisation des angles d'Euler pour la commande introduit des singularités supplémentaires. Il est donc préférable d'utiliser une représentation plus complète comme les quaternions. Dans [Guénard 04], la loi de commande pour la stabilisation de l'orientation est basée sur cette représentation et emploie une technique de backstepping.

Les autres types de tâches sont la stabilisation en vitesse, la stabilisation en une position fixe et le suivi de trajectoire. La stabilisation en vitesse consiste à définir une orientation désirée à un régulateur d'orientation et à s'assurer que les conditions de vol en régime quasi-stationnaire sont respectées. Dans [Guénard 08], la commande en vitesse est basée sur une technique de backstepping. Cette commande peut être alimentée par une consigne d'asservissement visuel comme il est proposé dans [Hamel 02a, Guénard 07]. La consigne de la commande en position peut alimenter la commande en vitesse. Les différentes commandes (position, vitesse, orientation) sont alors organisées en plusieurs boucles imbriquées comme dans [Bourquardez 09] où la consigne en position provient d'un schéma d'asservissement visuel.

Le suivi de trajectoire peut être basé sur des techniques de backstepping [Mahony 02], des modèles prédictifs de commande [Kim 03] ou des lois de commande par retour d'état non linéaire [Hua]. Ces techniques reposent sur l'hypothèse que l'état complet du drone peut être estimé. En pratique, certaines composantes du vecteur d'état ne sont pas disponibles comme la vitesse de translation par exemple. Afin de résoudre ce problème, une première solution consiste à définir un observateur afin d'estimer les composantes manquantes. Une autre solution est proposée dans [Bertrand 07] et consiste à construire une loi de commande par retour d'état partiel lorsque la vitesse de translation n'est pas disponible.

Le suivi de chemin visuel a été abordé pour les drones quadrirotors dans [Chitrakaran 06, Neff 07]. Dans ces travaux, la caméra embarquée est munie de deux degrés de liberté supplémentaires (caméra pan-tilt) afin de la rendre totalement actionnée. L'objectif de commande est une nouvelle fois formulé comme une tâche de positionnement de la caméra et est réalisée par asservissement visuel.

Nous définirons l'objectif de commande pour ce type de robot dans la Section 3.4. Des résultats de simulations permettront d'étudier les performances de l'approche proposée (Section 3.5).

3.2 Cas des robots à roues non-holonomes

Soit $\Psi = \{\mathcal{I}_1,\ \mathcal{I}_2, \ldots \mathcal{I}_{n-1},\ \mathcal{I}_n\}$ un chemin sensoriel extrait de la mémoire MS comme défini dans le Chapitre 2. Nous nous intéressons ici au suivi de Ψ par un robot mobile à roues non-holonome. Dans un premier temps, nous précisons l'objectif de commande. Nous proposons ensuite un schéma de commande permettant de l'atteindre.

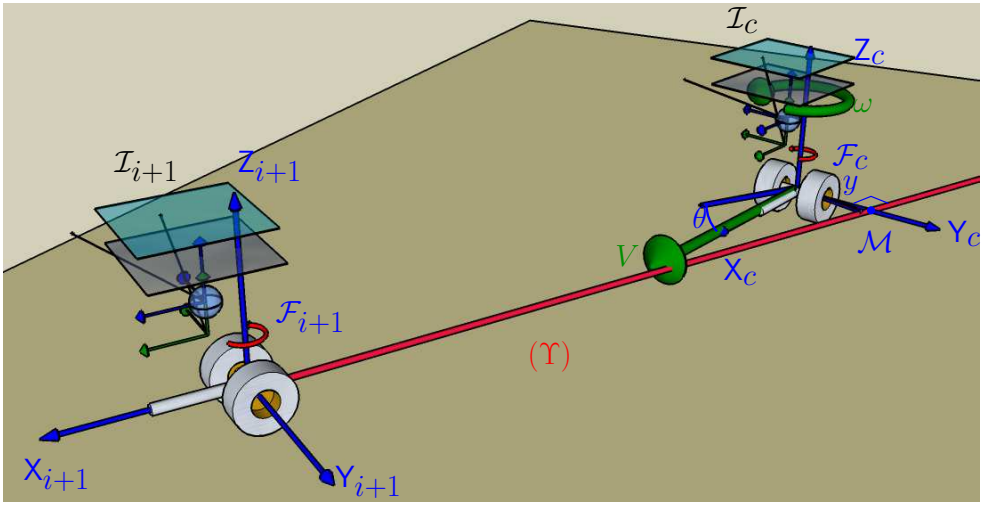
3.2.1 Objectif de commande

Le repère courant associé au véhicule est noté $\mathcal{F}_c = (\mathcal{O}_c, \mathsf{X}_c, \mathsf{Y}_c, \mathsf{Z}_c)$ où $\mathcal{O}_c$ est le centre de commande, X_c est dirigé dans la direction d'avance du véhicule, Z_c est dirigé perpendiculairement au sol et Y_c complète le trièdre direct. Soient $\mathcal{I}_i$ et $\mathcal{I}_{i+1}$ deux images consécutives du chemin sensoriel Ψ et $\mathcal{I}_c$ l'image courante (le capteur se situant entre les prises de vue $\mathcal{I}_i$ et $\mathcal{I}_{i+1}$). Nous supposons que les paramètres extrinsèques du capteur (c'est-à-dire la position et l'orientation du repère lié au capteur exprimées dans le repère de commande du robot) sont connus. On notera $\mathcal{F}_i = (\mathcal{O}_i, \mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i)$, $\mathcal{F}_{i+1} = (\mathcal{O}_{i+1}, \mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})$ et $\mathcal{F}_c$ les repères associés au véhicule lors des prises de vue $\mathcal{I}_i$, $\mathcal{I}_{i+1}$ et $\mathcal{I}_c$.

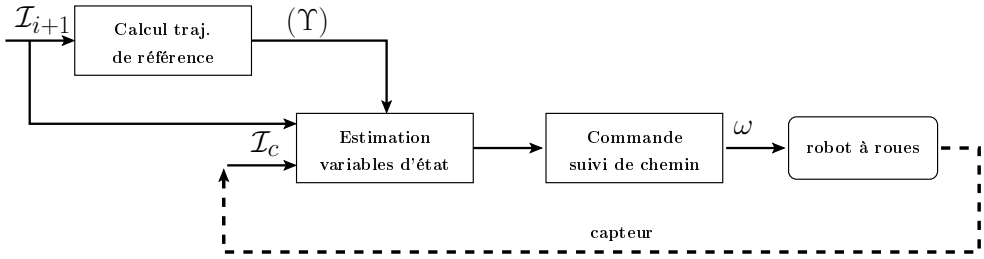
L'objectif de commande est d'amener successivement le capteur, rigidement fixé au corps du robot, aux positions et aux orientations correspondant aux prises de vue définissant le chemin sensoriel Ψ.

À cette fin, nous utiliserons deux stratégies différentes. La première stratégie consiste à guider $\mathcal{I}_c$ vers $\mathcal{I}_{i+1}$ en régulant asymptotiquement l'axe X_c sur la droite $(\Upsilon) = (\mathcal{O}_{i+1}, \mathsf{X}_{i+1})$ (voir Fig. 3.2). L'objectif de commande est alors atteint si l'axe d'avance courant X_c est régulé sur (Υ) avant que l'origine de $\mathcal{F}_c$ atteigne l'origine de $\mathcal{F}_{i+1}$. Nous verrons dans la Section 3.2.3.2 que cette stratégie a l'avantage d'être d'une mise en œuvre simple. Cependant, la trajectoire de référence est dans ce cas discontinue lors des changements d'image clé.

La seconde stratégie consiste à définir une trajectoire de référence (Υ) continue. Afin d'atteindre l'objectif de commande, (Υ) doit passer par les origines de $\mathcal{F}_i$ et $\mathcal{F}_{i+1}$.

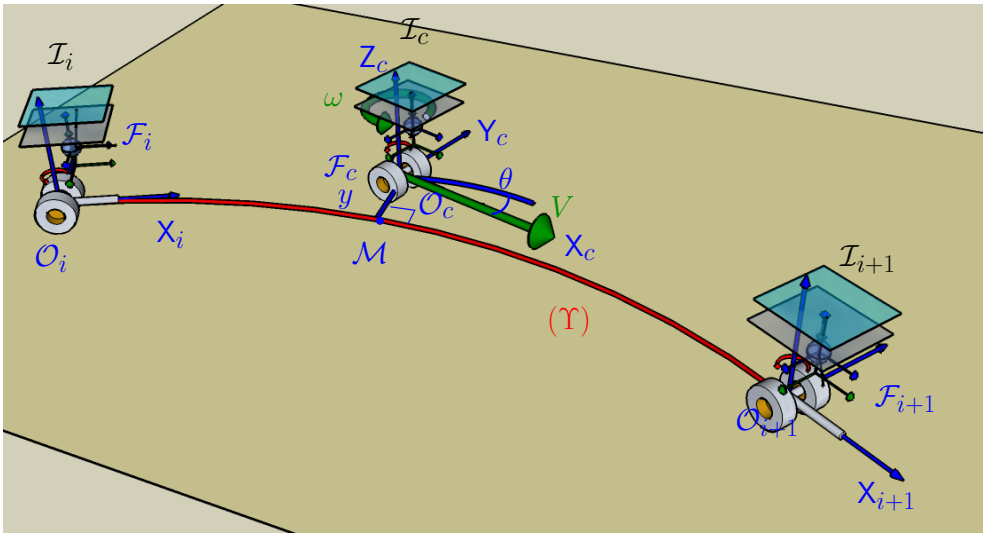


(a) Repères

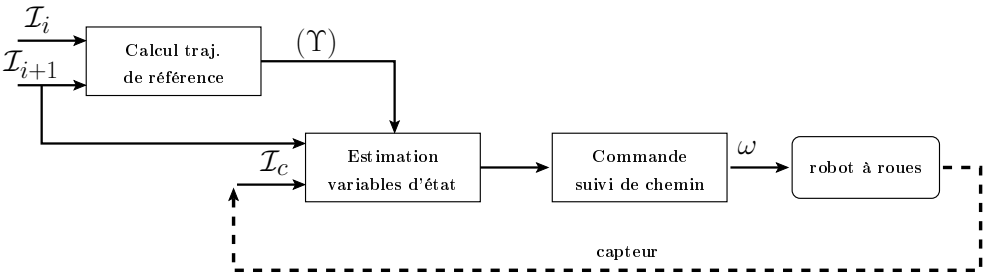


(b) Schéma bloc de la commande

FIG. 3.2 – Réalisation de l'objectif de commande par la Stratégie 1.



(a) Repères



(b) Schéma bloc de la commande

FIG. 3.3 – Réalisation de l'objectif de commande par la Stratégie 2.

De plus, les droites $(\mathcal{O}_i, \mathsf{X}_i)$ et $(\mathcal{O}_{i+1}, \mathsf{X}_{i+1})$ doivent être tangentes à (Υ) en $\mathcal{O}_i$ et $\mathcal{O}_{i+1}$ (voir Fig. 3.3). L'objectif de commande se ramène alors à la régulation de X_c le long de cette trajectoire avant que l'origine de $\mathcal{F}_c$ atteigne l'origine de $\mathcal{F}_{i+1}$. Par construction, les discontinuités dans la trajectoire de référence disparaissent au prix d'une complexité de mise en œuvre accrue comme nous le verrons dans le paragraphe 3.2.3.2. De manière assez classique, nous utiliserons des splines cubiques de Hermite pour construire (Υ). On note que nous ne cherchons ici qu'à éliminer les problèmes de discontinuité dans la trajectoire de référence.

Assez clairement, ces deux stratégies (Stratégies 1 et 2) se ramènent à des tâches de suivi du chemin de référence (Υ). Pour résoudre ce problème, nous avons choisi de concevoir des lois de commande basées sur le formalisme des systèmes chaînés. Comme nous le verrons dans la suite, ce choix permet de répondre à l'ensemble de nos objectifs :

- le capteur peut avoir les mêmes contraintes de déplacement que le robot,
- le robot se déplace à une vitesse longitudinale donnée,

– la trajectoire parcourue est maîtrisée.

La définition de la loi de commande est réalisée en deux étapes. Tout d'abord, le modèle d'état du robot est converti en un système chaîné ce qui permet d'obtenir une expression linéaire. Les concepts de l'automatique linéaire peuvent ensuite être exploités pour construire une loi de commande avec des performances garanties (voir Section 3.2.3).

3.2.2 Modélisation des robots à roues

Nous supposons que les véhicules à roues sont des corps rigides, symétriques, se déplaçant sur des sols plans horizontaux et à vitesse réduite. Il est ainsi possible de négliger les composantes dynamiques (suspensions...) et la déformation des pneumatiques. En outre, le contact entre la roue et le sol est supposé sans glissement. Compte tenu de ces hypothèses, les lois de commande utilisées pour les véhicules à roues se baseront sur un modèle cinématique. Parmi les types de robots mobiles à roues, on rencontre principalement ceux constitués de deux roues arrières non orientables, actionnées indépendamment l'une de l'autre, et d'une roue folle placée à l'avant et ceux constitué de deux roues arrières non orientables et de deux roues avant orientables.

Les modèles cinématiques de ces deux types de robots mobiles (char et bicyclette) sont présentés dans la suite.

Modèle char L'état du véhicule peut être défini dans le repère de Frenet relatif à la trajectoire de référence (Υ) à suivre. On introduit les notations suivantes (se référer à la Figure 3.4) :

- $\mathcal{O}_c$ est le centre de l'essieu arrière (point de contrôle).
- $\mathcal{M}$ est le point appartenant à (Υ) le plus proche de $\mathcal{O}_c$. Ce point est supposé unique, ce qui est réaliste en pratique si la trajectoire est suffisamment continue et si le véhicule reste proche de celle-ci.
- s est l'abscisse curviligne du point $\mathcal{M}$ le long de (Υ) et $c(s)$ définit la courbure de (Υ) en ce point.
- y et θ sont respectivement les erreurs latérale et angulaire du véhicule par rapport au chemin (Υ).
- $\dot{\phi_G}$ et $\dot{\phi_D}$ sont les vitesses angulaires des roues gauche et droite.
- V est la vitesse longitudinale (vitesse de $\mathcal{O}_c$ le long de l'axe X_c du repère $\mathcal{F}_c$).
- ω est la vitesse angulaire autour de l'axe Z_c du repère $\mathcal{F}_c$.

La configuration du véhicule peut être décrite sans ambiguïté par le vecteur d'état $\mathsf{e} = [s\ y\ \theta]^\top$: les deux premières composantes représentent la position du centre du robot tandis que la troisième composante décrit sa direction. Le vecteur de commande d'un tel robot est $[\dot{\phi_D}\ \dot{\phi_G}]^\top$. Par une relation inversible, il est possible de passer du vecteur de commande au vecteur de commande virtuel $[V\ \omega]^\top$:

$$\begin{pmatrix} V \\ \omega \end{pmatrix} = \frac{R}{2} \begin{pmatrix} 1 & 1 \\ -1/L & 1/L \end{pmatrix} \begin{pmatrix} \dot{\phi_D} \\ \dot{\phi_G} \end{pmatrix}$$

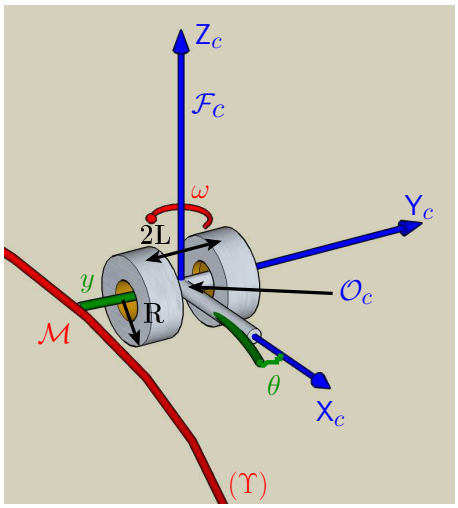


FIG. 3.4 – Modèle char pour le suivi de chemin.

avec R le rayon des roues et $2L$ la largeur de l'essieu arrière.

Le modèle cinématique d'un tel robot est obtenu en écrivant que les vecteurs vitesses au point $\mathcal{O}_c$ et aux milieux des roues arrières sont dirigés dans le plan des roues et que le mouvement du véhicule est, à chaque instant, une rotation autour du centre instantané de rotation. Ce modèle s'écrit alors :

$$\begin{cases} \dot{s} = \dfrac{V\cos\theta}{1-yc(s)} \\ \dot{y} = V\sin\theta \\ \dot{\theta} = \omega - \dfrac{Vc(s)\cos\theta}{1-yc(s)} \end{cases} \tag{3.1}$$

Le modèle (3.1) est clairement singulier si $y = \dfrac{1}{c(s)}$ c'est-à-dire quand le point $\mathcal{O}_c$ est superposé avec le centre de courbure d'abscisse s. Cependant, cette configuration n'est jamais rencontrée en pratique car, d'une part, la courbure de la trajectoire de référence (Υ) est généralement assez faible et, d'autre part, le véhicule est supposé proche de (Υ).

Modèle bicyclette Ce type de véhicule peut se ramener à une roue arrière motrice et à une roue avant directrice. On introduit les notations supplémentaires suivantes (voir Fig. 3.5) :

- δ est l'angle de braquage virtuel des roues avant (moyenne des angles de braquage droit et gauche).
- l est l'empattement (la distance entre les centres des essieux arrière et avant).

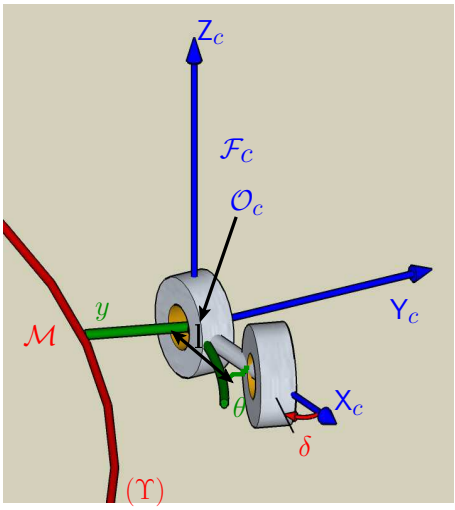


FIG. 3.5 – Modèle bicyclette pour le suivi de chemin.

Le vecteur de commande virtuel de ce modèle est $[V\ \delta]^\top$. Lorsqu'on suppose que la vitesse longitudinale est non nulle ($V \neq 0$), le modèle bicyclette est équivalent au modèle char et la vitesse angulaire ω est liée à l'angle de braquage par la relation :

$$\omega = \frac{V \tan\delta}{l} \tag{3.2}$$

L'équation d'état du modèle bicyclette (également appelée modèle d'Ackermann) peut alors être obtenu à partir de (3.1) et (3.2) :

$$\begin{cases} \dot{s} = \dfrac{V\cos\theta}{1 - yc(s)} \\ \dot{y} = V\sin\theta \\ \dot{\theta} = \dfrac{V\tan\delta}{l} - \dfrac{Vc(s)\cos\theta}{1 - yc(s)} \end{cases} \tag{3.3}$$

On notera que les singularités du modèle (3.3) sont identiques à celles du modèle char.

3.2.3 Commande

Sans perte de généralité, nous utilisons le modèle char décrit par l'équation (3.1). Il a été prouvé dans [Samson 95] qu'un système non linéaire comme (3.1) peut être converti par des changements de variable en un système dit *chaîné*, beaucoup plus simple à manipuler du point de vue de la commande. Dans le paragraphe 3.2.3.1, l'équation (3.1) est convertie en un système chaîné. La loi de commande est ensuite construite à partir de ce système en exploitant des outils classiques d'automatique linéaire. Nous verrons alors que les trajectoires spatiales peuvent être commandées quelle que soit la vitesse longitudinale [Thuilot 04].

3.2.3.1 Chaînage du système

Soient la transformation d'état donnée par (3.4) et la transformation de commande donnée par (3.5) :

$$\begin{aligned}\Phi(\begin{bmatrix} s & y & \theta \end{bmatrix}) &= \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \\ &\triangleq \begin{bmatrix} s & y & (1-yc(s))\tan\theta \end{bmatrix} \end{aligned} \tag{3.4}$$

$$(m_1, m_2) = \Xi(V, \delta) \tag{3.5}$$

avec :

$$m_1 \triangleq V\frac{\cos\theta}{1-yc(s)} \tag{3.6}$$

$$m_2 \triangleq \frac{d}{dt}((1-yc(s))\tan\theta) \tag{3.7}$$

Les transformations Φ et Ξ sont inversibles sous les conditions $y \neq \dfrac{1}{c(s)}$ et $V \neq 0$ et sous la condition $\theta \neq \frac{\pi}{2}[\pi]$. Cette configuration correspond à une direction du véhicule perpendiculaire à la trajectoire. Dans la pratique, on considérera que l'écart angulaire restera dans l'intervalle $]-\frac{\pi}{2};\frac{\pi}{2}[$.

En substituant (3.4), (3.6) et (3.7) dans (3.1), le système non linéaire (3.1) peut être réécrit sans approximation sous la forme chaînée suivante :

$$\begin{cases} \dot{a}_1 = m_1 \\ \dot{a}_2 = a_3\, m_1 \\ \dot{a}_3 = m_2 \end{cases} \tag{3.8}$$

Le modèle (3.8) constitue un système chaîné, dépendant de deux variables de commande : m_1 qui est homogène à la vitesse d'avance ($\dot{s}$) du véhicule le long de (Υ) et m_2 dépendant de la vitesse de rotation ω (la dépendance de m_2 à ω intervient par l'intermédiaire de $\dot{\theta}$ dans (3.1)).

Afin de construire une loi de commande avec des performances indépendantes de la vitesse d'avance V, il est judicieux de réécrire le système chaîné (3.8) en dérivant par rapport à l'abscisse curviligne plutôt que par rapport au temps. En notant : $\acute{a_i} = \frac{da_i}{da_1}$, la forme chaînée (3.8) s'écrit alors :

$$\begin{cases} \acute{a_1} = 1 \\ \acute{a_2} = a_3 \\ \acute{a_3} = \frac{m_2}{m_1} = m_3 \end{cases} \tag{3.9}$$

3.2.3.2 Loi de commande

En utilisant le système linéarisé exact (3.9), une loi de commande dédiée au suivi du chemin (Υ) peut être construite de la façon suivante. L'objectif du suivi de chemin est d'asservir à zéro les erreurs latérale et angulaire par rapport à (Υ). D'après (3.4),

cet objectif se traduit par la convergence des variables a_2 et a_3 vers zéro. Au vu de la structure du système (3.9) (équivalent à un double intégrateur), cela peut être obtenue par le choix d'un régulateur proportionnel dérivé pour la variable m_3 comme suit :

$$m_3 = -K_d a_3 - K_p a_2 \qquad (K_p,\ K_d) \in \mathcal{R}^{+*} \times \mathcal{R}^{+*} \tag{3.10}$$

où K_d et K_p sont deux réels définis positifs fixant la réponse du système. Ce régulateur permet, en effet, d'aboutir à l'équation différentielle suivante :

$$\frac{da_2'}{da_1} + K_d a_2' + K_p a_2 = 0. \tag{3.11}$$

Les gains K_d et K_p sont donc homogènes à des gains proportionnel et dérivé d'un correcteur classique. De plus, comme $a_2 = y$, les gains K_d et K_p imposent une distance d'établissement. Par conséquent, pour une erreur latérale initiale donnée, la trajectoire du véhicule est identique quelle que soit la valeur de V, même si cette vitesse, non nulle, varie avec le temps [Thuilot 04]. Par inversion de la transformation chaînée, l'expression non linéaire de la loi de commande s'écrit :

$$\begin{array}{l} \omega(y,\theta) = V \left[\frac{\cos^3\theta}{(1-c(s)y)^2} \left(\frac{dc(s)}{ds} y \tan\theta \right.\right. \\ \quad -K_d(1-c(s)y)\tan\theta \\ \quad \left.\left. -K_p y + c(s)(1-c(s)y)\tan^2\theta \right) + \frac{c(s)\cos\theta}{1-c(s)y} \right] \end{array} \tag{3.12}$$

sous les trois conditions évoquées précédemment ($V \neq 0$, $y \neq \frac{1}{c(s)}$ et $\theta \neq \frac{\pi}{2}[\pi]$).

La loi de commande (3.12), valable pour les robots de type char peut être combinée avec l'équation (3.2) afin d'obtenir la loi de commande pour les systèmes de type bicyclette :

$$\begin{array}{l} \delta(y,\theta) = \arctan\left(l \left[\frac{\cos^3\theta}{(1-c(s)y)^2} \left(\frac{dc(s)}{ds} y \tan\theta \right.\right.\right. \\ \quad -K_d(1-c(s)y)\tan\theta \\ \quad \left.\left.\left. -K_p y + c(s)(1-c(s)y)\tan^2\theta \right) + \frac{c(s)\cos\theta}{1-c(s)y} \right] \right) \end{array} \tag{3.13}$$

Comme pour la loi de commande (3.12), la loi (3.13) suppose que la vitesse longitudinale est non nulle ($V \neq 0$) et que les hypothèses $y \neq \frac{1}{c(s)}$ et $\theta \neq \frac{\pi}{2}[\pi]$ sont vérifiées.

Réglage des gains Nous nous intéressons maintenant au réglage des gains de commande K_p et K_d. La relation (3.11) définit un système du second ordre. On peut alors poser : $K_p = \omega_n^2$ et $K_d = 2\zeta\omega_n$ où ω_n représente la pulsation propre du système et ζ le facteur d'amortissement. Afin d'éviter toute oscillation de l'écart latéral autour de zéro et garantir la convergence la plus rapide possible, le système doit être amené en régime critique. Cette condition est réalisée si $\zeta = 1$. La solution à (3.11) est donnée dans ce cas par :

$$y(s) = y(s_0)\left(1 + \omega_n(s-s_0) + y'(s_0)(s-s_0)\right) e^{-\omega_n(s-s_0)}$$

où s_0 désigne l'abscisse curviligne initiale et $y(s_0)$ (respectivement $y'(s_0)$) l'écart latéral entre le chemin de référence et la position initiale du véhicule (respectivement sa dérivée en fonction de s). On considère que les objectifs de la régulation sont atteints dès que l'erreur latérale est inférieure à 5% de l'erreur initiale $y(s_0)$. La distance d'établissement d_m peut donc être définie par la relation :

$$0.05 = \left(1 + \omega_n d_m + y'(s_0) d_m\right) e^{-\omega_n d_m} \tag{3.14}$$

On suppose que lors du démarrage, l'évolution de l'erreur latérale est nulle ($y'(s_0) = 0$). On établit alors à l'aide d'un solveur numérique l'expression :

$$d_m \simeq \frac{4.75}{\omega_n} \tag{3.15}$$

Les valeurs des gains sont alors les suivantes :

$$K_p \simeq \frac{22.5}{d_m^2} \tag{3.16}$$

$$K_d \simeq \frac{9.5}{d_m} \tag{3.17}$$

Ces gains assurent que l'écart latéral aura diminué de 95% lorsque le véhicule aura parcouru une distance d_m le long de la trajectoire de référence.

Réalisation de nos objectifs Dans le cas général, il est nécessaire d'estimer le vecteur d'état du robot $\mathsf{e} = [s\ y\ \theta]^\top$ afin de calculer les commandes (3.12) et (3.13), l'abscisse curviligne s permettant d'estimer la courbure et sa dérivée au point $\mathcal{M}$ de (Υ). Nous verrons dans le Chapitre 4 comment une caméra peut permettre d'estimer ces paramètres.

Comme nous l'avons évoqué précédemment, la Stratégie 1 consiste à suivre le chemin $(\Upsilon) = (\mathcal{O}_{i+1}, \mathsf{X}_{i+1})$. Ce chemin ayant une courbure nulle ($c(s) = 0,\ \forall s$), les lois de commandes (3.12) et (3.13) se réécrivent sous les formes :

$$\omega(y, \theta) = V \cos^3\theta \left(-K_d \tan\theta - K_p y\right) \tag{3.18}$$

et

$$\delta(y, \theta) = \arctan\left(l \cos^3\theta \left[-K_d \tan\theta - K_p y\right]\right) \tag{3.19}$$

Dans ce cas, on peut se contenter de la mesure de l'état partiel $\mathsf{e_p} = [y\,\theta]^\top$ afin de calculer les commandes (3.18) et (3.19).

Jusque là, nous avons supposé que les variables d'état étaient parfaitement estimées à partir des données capteurs. Nous verrons dans le Chapitre 4 que, dans le cas d'une caméra, l'erreur angulaire est parfaitement estimée tandis que les erreurs de translation sont déterminées avec un facteur d'échelle $s \in \Re^{+*}$. Pour la Stratégie 1, la distance d'établissement réelle vaut alors $\frac{d_m}{s}$. On obtient à partir de (3.15) la relation suivante :

$$d_m \omega_n \simeq 4.75 s$$

D'après cette équation et (3.14), l'erreur latérale sera inférieure à 95% de l'erreur initiale $y(s_0)$ au bout d'une distance d_m si le facteur d'échelle est surestimé. Pour la Stratégie 2, nous n'avons pas démontré que le suivi sera réalisé de façon convenable avec un facteur d'échelle différent de 1. Nous verrons, cependant, lors des simulations (Section 3.3.2.1) que les résultats de la tâche de navigation sont satisfaisants lorsque s est supérieur à 1.

3.3 Résultats de simulation

Les simulations sont réalisées sous Matlab en considérant un robot d'intérieur non-holonome de type char muni d'une caméra fixe. Le repère caméra et le repère robot sont supposés confondus (l'axe optique de la caméra est dirigée vers le plafond). Le vecteur de translation pour passer du repère caméra au repère robot est donc nul : ${}^r\mathsf{t}_c = \mathsf{0}_{3\times1}$ et la matrice de rotation vaut l'identité : ${}^r\mathbf{R}_c = \mathsf{I}_{3\times3}$ (paramètres extrinsèques).

Rappelons que notre approche de navigation peut se décomposer en trois étapes : (1) construction de la mémoire sensorielle, (2) localisation initiale et (3) navigation autonome. Nous considérerons ici que la mémoire se résume à une séquence unique d'images. La localisation initiale sera la première image de la séquence et l'objectif consiste (sauf exception) à rejoindre la dernière image de la séquence.

Le chemin suivi lors de la phase d'apprentissage est représenté Fig. 3.6 dans le repère monde $\mathcal{F}_w$. Tout d'abord, le robot avance de la position $[2\ 8\ 0]^\top$ à la position $[7\ 8\ 0]^\top$ (l'orientation est nulle), il suit ensuite un arc de cercle de rayon 1 mètre dans le sens trigonométrique, de façon à arriver à la position $[8\ 7\ 0]^\top$ avec un angle de $-\frac{\pi}{2}$ radians. Le robot avance de 1 mètre (position $[8\ 6\ 0]^\top$). Le robot effectue ensuite un quart de cercle de 2 mètres de rayon, une translation de 4 mètres puis un demi-cercle de rayon 2 mètres afin de revenir à la position initiale. Le chemin total mesure 21 mètres.

Les positions du robot correspondant aux acquisitions des images de référence sont situées tous les 1 mètre au début du déplacement, puis tous les 0.5 mètre lors du dernier virage. La mémoire visuelle est alors formée de 27 images de référence. Le capteur simulé est une caméra qui acquiert des images de taille 640×480 pixels avec un objectif fisheye dont les paramètres intrinsèques sont ceux d'une caméra réelle : $f_u = 864.7$, $f_v = 831.8$, $u_0 = 305.1$ pixels, $v_0 = 266.9$ pixels et $\xi = 2.875$ (nous reviendrons sur ce modèle de caméra dans le Chapitre 4). La fréquence d'acquisition des images est fixée à 10 Hz. Nous considérons 200 points 3D coplanaires situés sur un plan horizontal situé à une hauteur de 3 mètres par rapport à l'origine du repère robot. Ces points sont projetés sur le plan image. L'appariement des points entre deux images est réalisé en simulation et permet d'estimer la matrice d'homographie en utilisant l'équation (4.14) définie p. 126. Les entrées de commande sont estimées à partir de la décomposition de cette matrice comme nous le verrons dans le Chapitre 4. On obtient ensuite aisément l'erreur angulaire θ et l'erreur latérale y (à un facteur d'échelle près). Pour la stratégie 2, l'abscisse curviligne est également nécessaire. Dans nos expérimentations, nous utilisons une spline de Hermite. Celle-ci est définie par l'équation paramétrée par le réel $t \in [0, 1]$:

$$\mathsf{M}(t) = h_{00}(t)\mathsf{M}_0 + h_{10}(t)\mathsf{T}_0 + h_{01}(t)\mathsf{M}_1 + h_{11}(t)\mathsf{T}_1 \tag{3.20}$$

avec $\mathsf{M}_0 = \mathsf{M}(0)$ et $\mathsf{M}_1 = \mathsf{M}(1)$ les positions initiale et finale et $\mathsf{T}_0 = \mathsf{T}(0)$ et $\mathsf{T}_1 = \mathsf{T}(1)$ les tangentes en ces points. Les coefficients du polynôme (3.20) sont donnés par :

$$\begin{cases} h_{00}(t) &= 2t^3 - 3t^2 + 1 \\ h_{10}(t) &= t^3 - 2t^2 + t \\ h_{01}(t) &= -2t^3 + 3t^2 \\ h_{11}(t) &= t^3 - t^2 \end{cases} \tag{3.21}$$

Les variables d'état nécessite de calculer l'estimation de la tangente et de la normale à la spline ainsi que de la courbure en un point $\mathcal{M}$ de la spline. Les coordonnées de ce point $\mathcal{M}$ sont $\mathsf{M}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$. La tangente et la normale sont respectivement :

$$\mathsf{T}(t) = \frac{1}{x'^2(t) + y'^2(t)} \begin{pmatrix} x'^2(t) \\ y'^2(t) \end{pmatrix}$$

et

$$\mathsf{N}(t) = \frac{1}{x'^2(t) + y'^2(t)} \begin{pmatrix} -y'^2(t) \\ x'^2(t) \end{pmatrix}$$

avec $x^{'}(t)$ et $y^{'}(t)$ les dérivées de $x(t)$ et $y(t)$ par rapport à t. La courbure en ce point est définie en abscisse curviligne par :

$$c(s(t)) = \frac{x^{'}(t)y^{''}(t) - x^{''}(t)y^{'}(t)}{(x'^2(t) + y'^2(t))^{\frac{3}{2}}}$$

Les paramètres de la spline cubique de Hermite utilisée dans ce cas sont obtenus à partir de l'erreur angulaire, de l'erreur latérale et de l'abscisse curviligne calculées pour l'image clé précédente.

Nous définissons l'erreur dans l'image ErrImage comme la moyenne des distances entre les positions des points appareillés.

Dans un premier temps, nous supposons que les paramètres extrinsèques et intrinsèques de la caméra sont parfaitement connus, que les données capteur sont non bruitées et que le facteur d'échelle est connu de manière exacte. Nous analysons alors l'influence des paramètres intervenant dans la commande (critère de changement d'image clé, amplitude des erreurs initiales, stratégie de suivi et distance d'établissement). Nous analysons ensuite l'influence d'erreurs sur les paramètres relatifs à l'estimation des variables d'état (facteur d'échelle et paramètres extrinsèques du capteur). Nous proposons enfin d'analyser la robustesse de l'approche vis-à-vis d'erreurs sur les paramètres intrinsèques du capteur visuel et en présence de bruits de mesure.

3.3.1 Influence des paramètres intervenant dans la commande

Dans cette partie, nous étudions l'influence des critères de changement d'image clé, des erreurs initiales, des stratégies de commande employées et de la distance d'établissement sur le suivi du chemin sensoriel.

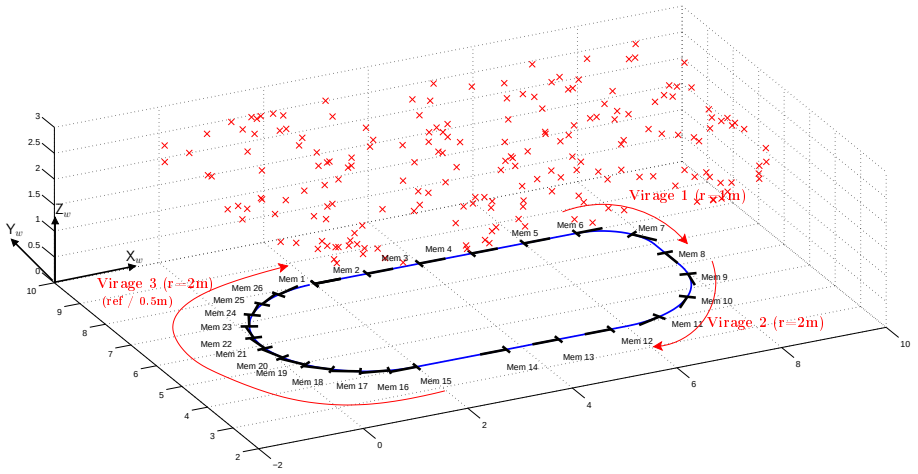


FIG. 3.6 – Environnement de simulation utilisé pour la navigation des robots à roues. Les coordonnées sont exprimées en mètres. Les points 3D permettant d'estimer l'état sont représentés par des croix.

3.3.1.1 Influence du choix du critère de changement d'image clé

Deux critères de changement d'image clé sont étudiés ici :

Critère 1 : ce critère est défini dans l'espace capteur : une image clé est atteinte lorsque l'erreur dans l'image ErrImage est inférieure à un seuil.

Critère 2 : ce critère est défini dans l'espace cartésien : une image clé est atteinte lorsque l'abscisse curviligne du point $\mathcal{M}$ de (Υ) le plus proche du centre du robot est supérieure à l'abscisse curviligne de $\mathcal{O}_{i+1}$.

Nous utilisons la Stratégie 2 et analysons l'influence du critère de changement d'image sur la réalisation de l'objectif de suivi. La distance d'établissement d_m est fixée à 1 mètre (distance raisonnable au vue des dimensions de l'environnement) et la vitesse d'avance V à 0.2 m/s. La position initiale du robot est $[2\ 8.1\ 0]^\top$ ($y(s_0) = 0.1$ m) et l'orientation est nulle[2] (donc $\theta(s_0) = 0$). L'erreur initiale dans l'image est de 40 pixels. Nous considérons le Critère 1 avec un seuil fixé à 3 pixels. Les résultats sont représentés Figure 3.7 où les différentes grandeurs (commande, écarts, erreur dans l'image) sont données en fonction du temps (exprimé en secondes) et les croix marquent un changement d'image clé.

On observe sur la Fig. 3.7 (a) que les résultats du suivi de chemin sont satisfaisants. La moyenne de l'écart latéral réel par rapport au chemin parcouru lors de l'apprentissage est nulle et l'écart type est de 1 centimètre [3]. Les erreurs angulaire et latérale convergent vers zéro (voir Fig. 3.7 (c)) tandis que l'erreur dans l'image décroît vers le

[2] Si cela n'est pas précisé, cela sera toujours le cas lors des simulations suivantes.

[3] Nous considérons que les écarts par rapport au chemin appris sont mesurés une fois que l'image Mem2 a été rejoint afin de ne pas prendre en compte la phase initiale pour rejoindre le chemin appris.

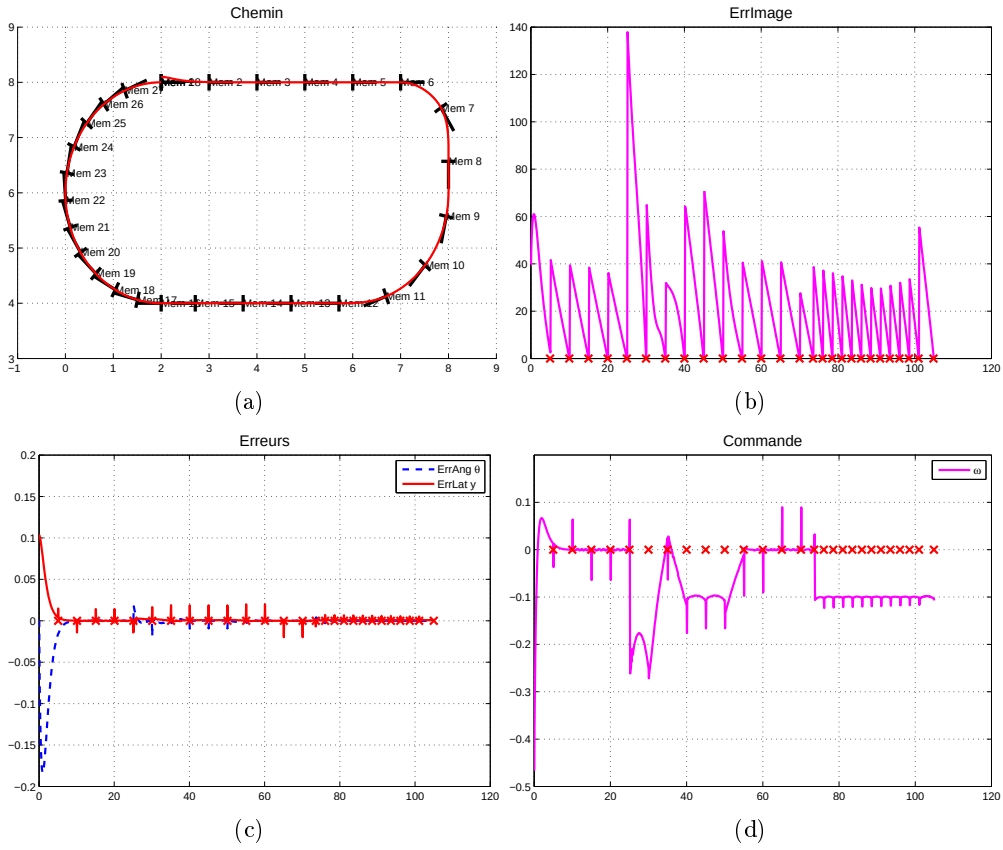
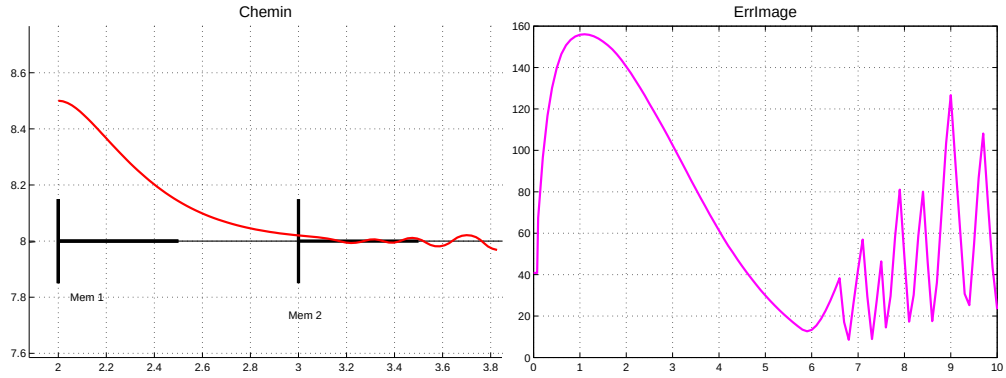


FIG. 3.7 – Critère 1 (ErrImage $<$ 3 pixels) avec une position initiale proche du chemin appris ($y(s_0)$= 0.1 m). (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels), (c) erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et (d) commande ω (exprimée en rad/s) en fonction du temps (en secondes).



(a) (b)

FIG. 3.8 – Critère 1 (ErrImage < 3 pixels) avec une position initiale éloignée du chemin appris ($y(s_0) = 0.5$ m). (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels) en fonction du temps (en secondes).

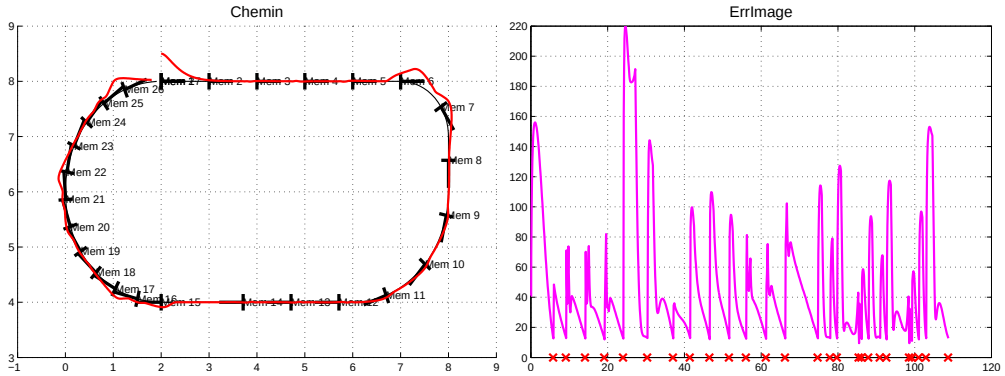
seuil fixé pour chaque image du chemin sensoriel (voir Fig. 3.7 (b)).

Nous positionnons maintenant le robot en $[2\ 8.5\ 0]^\top$ (l'erreur latérale initiale vaut alors 0.5 mètre). Les résultats de la navigation autonome sont représentés Figure 3.8. Le robot se rapproche du chemin appris avec les caractéristiques voulues ($y(s) \leq 0.05y(s_0)$ après avoir parcouru une distance d_m). Cependant, cela ne permet pas d'atteindre le seuil défini dans l'espace capteur (ErrImage < 3 pixels ; voir Fig. 3.8 (b)). La tâche de suivi échoue donc.

Afin de surmonter cette difficulté, il est possible d'augmenter le seuil associé au Critère 1. Le robot étant initialement positionné comme précédemment ($[2\ 8.5\ 0]^\top$), nous fixons le seuil du Critère 1 à 13 pixels. Les résultats sont représentés Fig. 3.9. On observe que l'erreur entre le chemin appris et le chemin parcouru n'est plus régulée de façon convenable dans l'espace cartésien : l'erreur moyenne est de 5 centimètres avec un écart type de 6 centimètres. Le plus souvent, ce comportement résulte du changement d'image qui se produit bien avant que la position correspondant à la prise de vue de l'image de référence ne soit atteinte.

Nous utilisons maintenant le Critère 2. Les résultats sont représentés Fig. 3.10. Le comportement de l'erreur dans l'image et le chemin parcouru dans l'espace cartésien (écart latéral moyen nul avec un écart type inférieur à 1 centimètre) sont satisfaisants bien que l'erreur latérale initiale soit importante. Dans ce cas, le Critère 2 semble donc plus adapté que le Critère 1.

Au vu de ces résultats, plusieurs choix sont envisageables. Un bon comportement peut être obtenu avec le Critère 2, même en présence d'erreurs initiales importantes. Cependant, dans le cadre de la Stratégie 1, cette solution nécessite d'estimer l'abscisse



(a) (b)

FIG. 3.9 – Critère 1 (ErrImage < 13 pixels) avec une position initiale éloignée du chemin appris ($y(s_0) = 0.5$ m). (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels) en fonction du temps (en secondes).

curviligne en plus des variables d'état. Si les erreurs initiales sont faibles, le Critère 1 peut être employé. Dans le cadre de la Stratégie 1, cela impose un échantillonnage plus fin dans les virages serrés afin d'assurer que les erreurs soient faibles lors des changements d'image clé. Une autre possibilité, que nous n'avons pas approfondie dans cette thèse, est de définir un critère dans l'image fonction de l'erreur initiale. Cela permettrait de traiter de manière indifférente les deux stratégies, même en présence d'erreurs initiales importantes.

3.3.1.2 Influence des erreurs initiales

Nous proposons ici d'étudier la robustesse de notre approche vis-à-vis d'erreurs initiales importantes (erreur latérale tout d'abord puis erreur angulaire ensuite).

Erreur latérale Nous vérifions que la commande par suivi de chemin et le critère de changement d'image clé permettent de réaliser l'objectif de commande par la Stratégie 1 pour cinq positions initiales situées entre $[2\ 7\ 0]^\top$ et $[2\ 9\ 0]^\top$. L'orientation initiale est identique à celle du chemin appris au niveau de l'image désirée ($\theta(s_0) = 0$). Les chemins parcourus sont représentés Fig. 3.11 (a). La distance d'établissement étant plus importante que la distance pour rejoindre l'image Mem2, l'écart latéral diminue mais n'est pas régulé à zéro avant d'atteindre cette image clé.

Nous vérifions maintenant que les performances attendues en terme de convergence sont atteintes : la valeur d'écart latéral $y = 0.05y(s_0)$ est atteinte pour une distance inférieure à 1 mètre (distance d'établissement) sur le chemin à suivre (0.97 mètre pour $|y(s_0)| = 1$ mètre et 0.96 mètre pour $|y(s_0)| = 0.5$ mètre).

Les différentes grandeurs (commande, erreurs latérale et angulaire, erreur dans l'image) sont représentées Fig. 3.11 (b), (c) et (d) pour une position initiale située à 1 mètre du chemin appris.

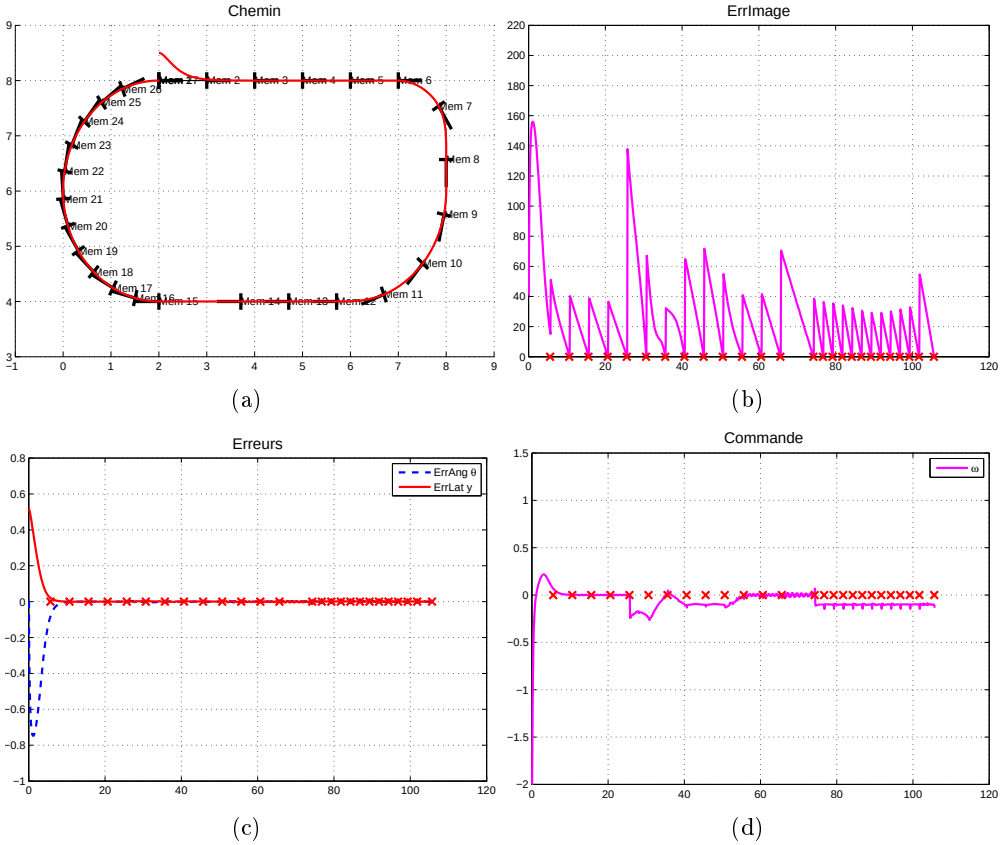


FIG. 3.10 – Critère 2 avec une position initiale éloignée du chemin appris. (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels), (c) erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et (d) commande ω (exprimée en rad/s) en fonction du temps (en secondes).

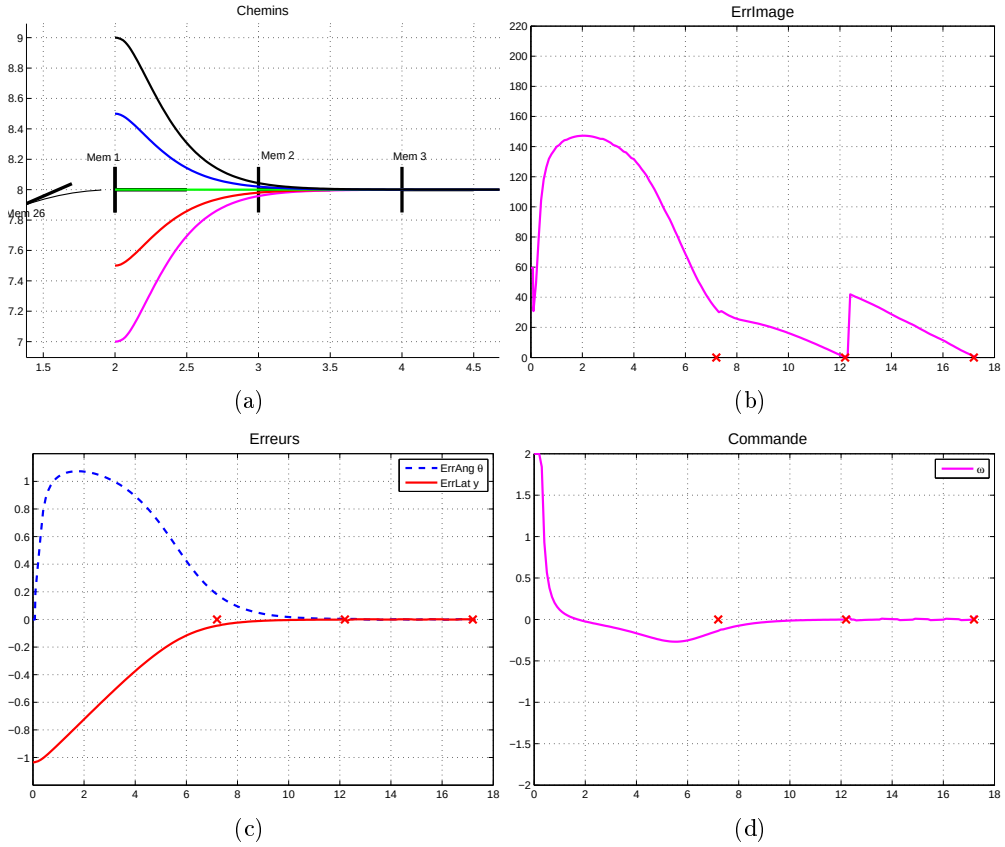


FIG. 3.11 – Robustesse par rapport à l'écart latéral initial $y(s_0)$. (a) chemins parcourus pour des valeurs initiales de $y(s_0)$ valant -1 mètre, -0.5 mètre, 0 mètre, 0.5 mètre et 1 mètre (représentés en coordonnées métriques) et (b) erreur dans l'image (en pixels), (c) erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et (d) commande ω (exprimée en rad/s) en fonction du temps (en secondes) pour un écart latéral initial $y(s_0)$ = -1 mètre.

Erreur angulaire Nous analysons maintenant les performances de notre approche en présence d'une erreur angulaire initiale qui peut être importante. Le robot est positionné en $[2\ 8.5\ 0]^\top$ et l'orientation par rapport à la direction initiale du chemin appris (et donc l'écart angulaire initial $\theta(s_0)$) varie entre -20 degrés et 20 degrés. Les différents chemins parcourus sont représentés Fig. 3.12. Pour les différentes erreurs d'orientation initiales, la réalisation de la tâche de navigation est satisfaisante. Une fois encore, les résultats montrent qu'un écart latéral inférieur à 5% de l'écart initial est obtenu pour une distance inférieure ou égale à la distance d'établissement.

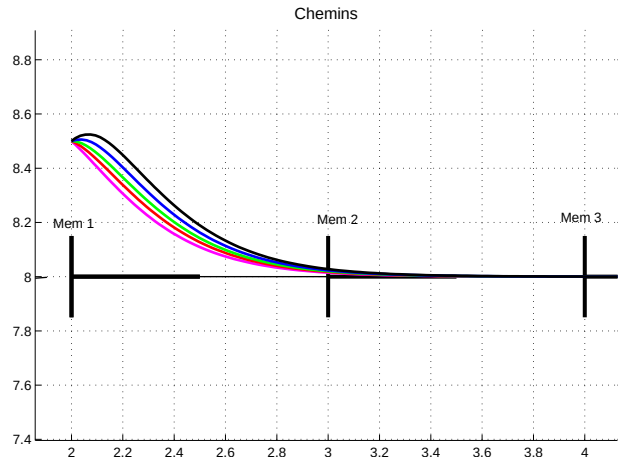


FIG. 3.12 – Chemins parcourus avec des valeurs initiales de l'état de $y(s_0) = 0.5$ mètre et $\theta(s_0)$ de -20 degrés, -10 degrés, 0 degré, 10 degrés et 20 degrés (représentés en coordonnées métriques).

3.3.1.3 Stratégie 1 et Stratégie 2

Nous comparons maintenant les Stratégies 1 et 2 le long du chemin complet. Le robot est initialement positionné en $[2\ 8.5\ 0]^\top$ avec une orientation nulle (on a donc $y(s_0) = 0.5$ mètre et $\theta(s_0) = 0$). Les chemins parcourus sont représentés Fig. 3.13. Dans les deux cas, le chemin parcouru lors de l'apprentissage est bien suivi : l'écart moyen par rapport au chemin appris est inférieur à 1 centimètre et atteint une valeur maximale d'environ 2 centimètres pour les deux stratégies.

Les évolutions de $[y\,\theta]$, de l'erreur dans l'image et de la commande sont représentées Fig. 3.14. Pour la Stratégie 1 (colonne de gauche), les discontinuités dans la référence dans les deux virages impliquent des discontinuités dans $[y\,\theta]$ (voir entre 25 secondes et 55 secondes puis entre 75 secondes et 105 secondes). Il en résulte des discontinuités dans la commande. La Stratégie 2 permet, quant à elle, d'obtenir des variations continues de $[y\,\theta]$.

Comme nous l'avons évoqué dans la Section 3.2.3.2, la Stratégie 1 a l'avantage d'être d'une mise en œuvre simple. Cependant, la trajectoire de référence est discontinue ce qui se traduit, comme nous venons de le voir, par des discontinuités dans la commande. Les discontinuités dans la trajectoire de référence disparaissent dans la Stratégie 2 au prix d'une complexité de mise en œuvre accrue.

3.3.1.4 Influence de la distance d'établissement

Dans ce paragraphe, on considère différentes valeurs de la distance d'établissement : 0.5 mètre, 1 mètre, 2 mètres et 5 mètres. On rappelle que les premières images clés du chemin sensoriel (Mem1 à Mem15) ont été acquises tous les 1 mètre puis les dernières

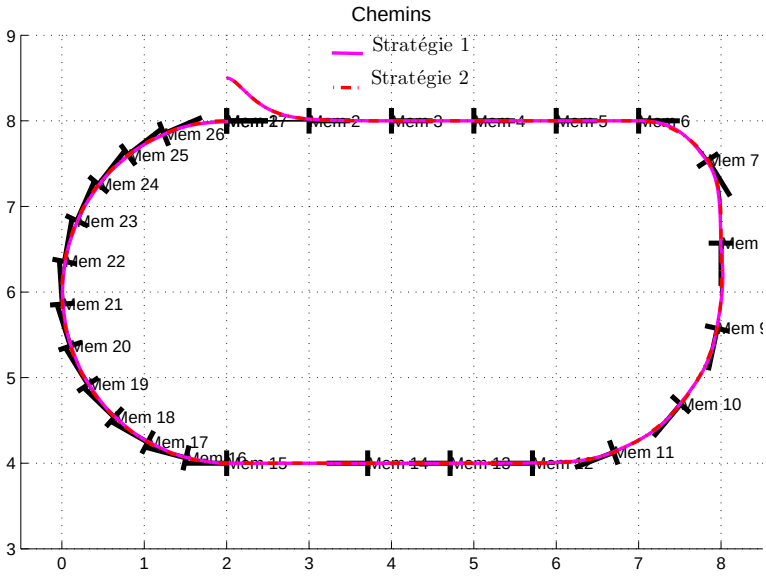


FIG. 3.13 – Chemins (représentés en coordonnées métriques) parcourus avec les Stratégies 1 et 2.

images (Mem16 à Mem26) tous les 0.5 mètre. L'influence de d_m sur la tâche de navigation est analysée dans la suite.

Stratégie 1 Les chemins parcourus en utilisant les différentes distances d'établissement sont représentés Fig. 3.15. Tant que cette distance d'établissement est inférieure ou égale à 2 mètres, le chemin appris est suivi avec un écart de moyenne inférieure à 1 centimètre et d'écart type inférieur à 2 centimètres. Avec une distance $d_m = 5$ mètres, on observe que le chemin parcouru se situe à l'extérieur des virages du chemin appris (l'écart moyen est alors de 17 centimètres et atteint 45 centimètres). Après environ 2.5 mètres parcourus en ligne droite, l'écart entre le chemin parcouru et le chemin appris diminue jusqu'à devenir nul.

Les évolutions de $[y\,\theta]$, de l'erreur dans l'image et de la commande avec des distances d'établissement de 0.5 mètre et 5 mètres sont représentés Figure 3.16. On observe sur cette figure qu'avec une distance $d_m = 0.5$ mètre, les vitesses atteignent des valeurs plus élevées (2 rad/s contre 0.18 rad/s en valeur absolue) et les sauts de vitesse sont beaucoup plus importants qu'avec une distance de 5 mètres. Une valeur importante de la distance d'établissement permet donc de s'éloigner des limites des actionneurs et d'améliorer le confort du matériel et/ou des passagers.

Stratégie 2 Les chemins parcourus sont représentés Fig. 3.17. Le chemin appris est bien suivi : l'écart moyen entre ce chemin et le chemin suivi est inférieur à 1 centimètre

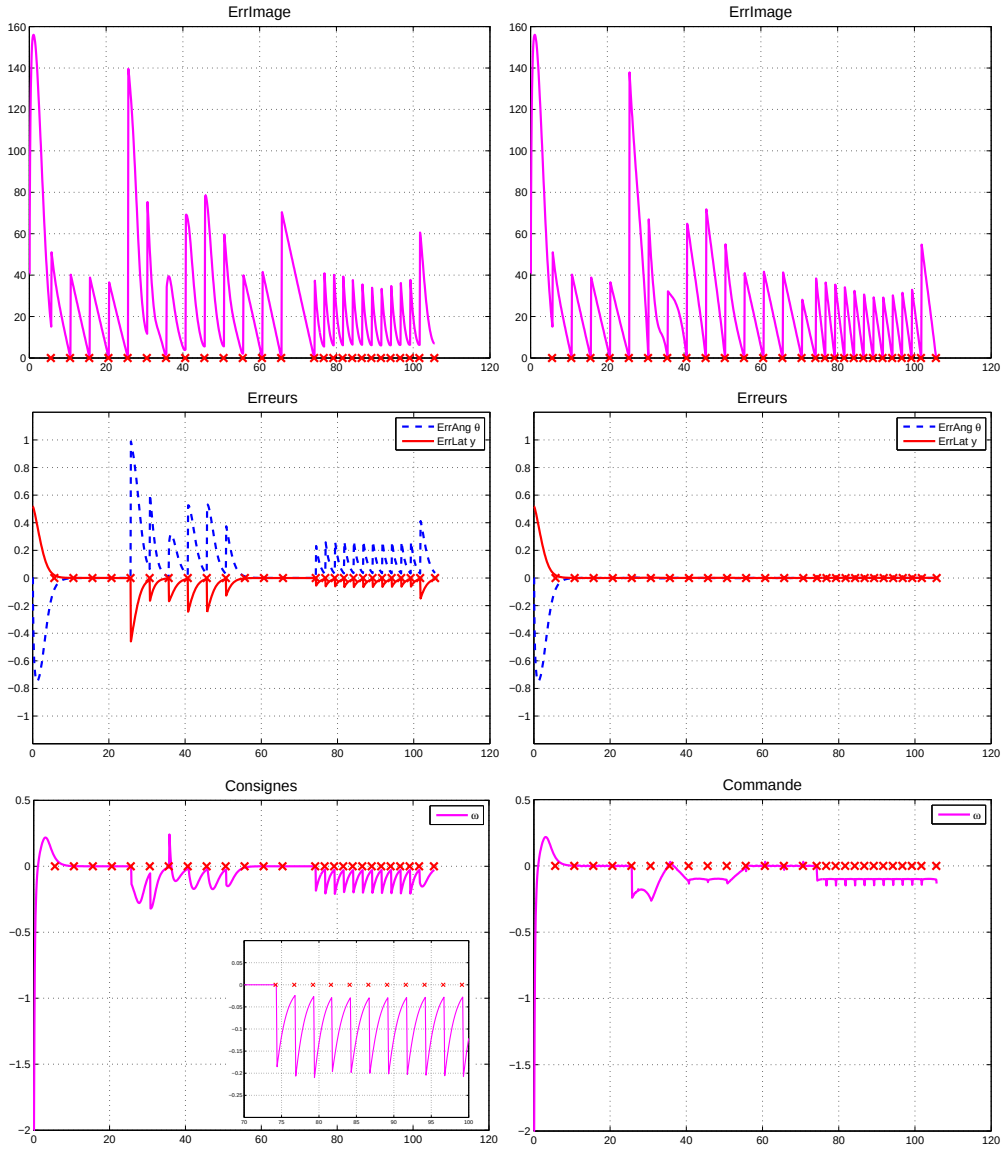


FIG. 3.14 – Erreur dans l'image (en pixels), erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et commande ω (exprimée en rad/s) en fonction du temps (en secondes) avec la Stratégie 1 (colonne de gauche) et avec la Stratégie 2 (colonne de droite).

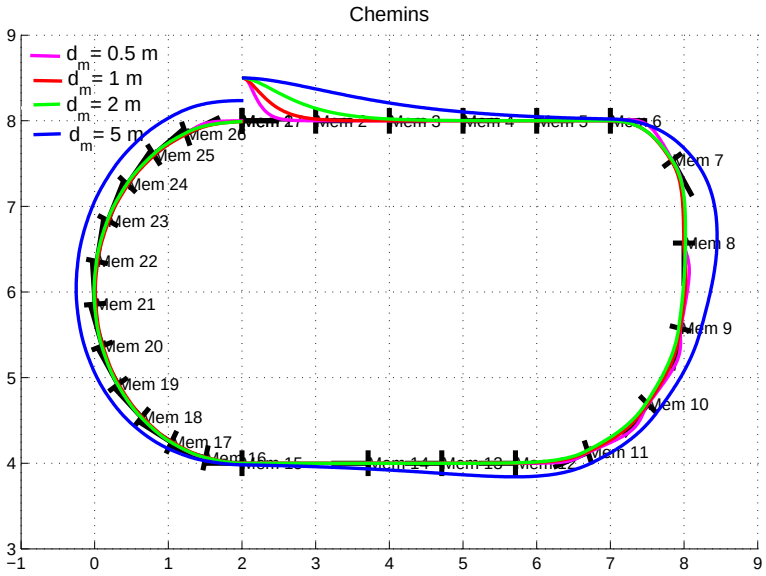


FIG. 3.15 – Chemins (représentés en coordonnées métriques) parcourus avec la Stratégie 1 et différentes distances d'établissement.

pour $d_m \leq 2$ mètres et vaut 4 centimètres pour $d_m = 5$ mètres. Contrairement à la Stratégie 1, le robot ne s'éloigne pas du chemin appris une fois qu'il a été rejoint, même lorsque la distance d'établissement est importante.

Comme nous venons de l'évoquer, il est préférable de choisir une valeur importante de la distance d'établissement afin d'éloigner le robot des limites de ses actionneurs et ainsi d'améliorer le confort du matériel et/ou des passagers. Dans la Stratégie 1, les erreurs initiales importantes dans les virages se traduisent par un écart latéral important par rapport à la trajectoire de référence. Cet écart est faible lorsque les positions d'acquisition des images clés sont proches (après Mem20) et croissent lorsqu'elles sont éloignées (vers Mem7 et Mem8). Afin de limiter l'amplitude des écarts latéraux par rapport à la trajectoire apprise, il serait donc souhaitable de sélectionner plus d'images clés dans les virages lors de la phase de construction de la carte sensorielle. Toutefois, cela entraînerait des changements de référence plus nombreux et un accroissement de la taille mémoire nécessaire au stockage de la mémoire sensorielle.

3.3.2 Influence des paramètres relatifs à l'estimation des variables d'état

La distance d'établissement est maintenant fixée à $d_m = 1$ mètre et nous nous focalisons sur l'influence des paramètres relatifs à l'estimation des variables d'état c'est-à-dire le facteur d'échelle et les erreurs sur les paramètres extrinsèques du capteur.

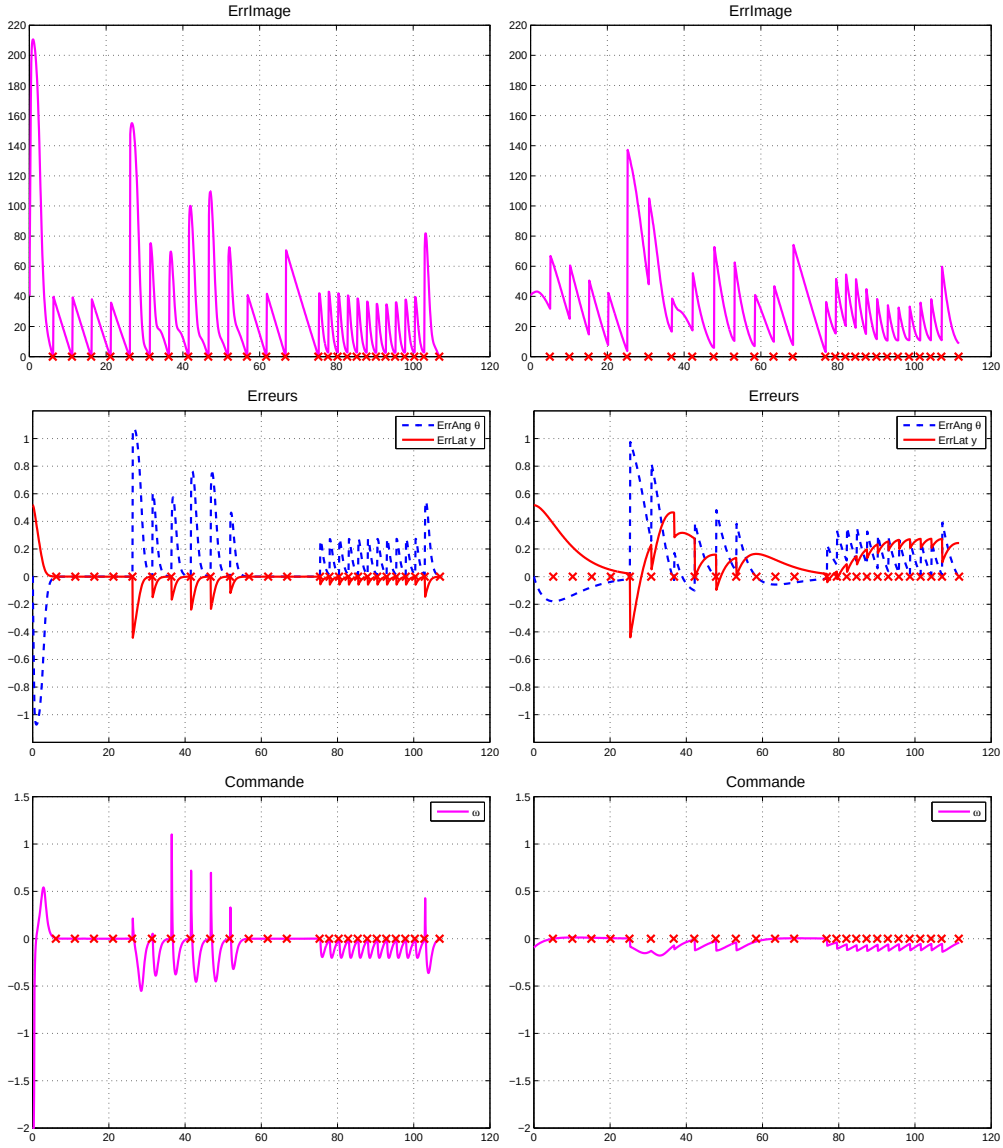


FIG. 3.16 – Erreur dans l'image (en pixels), erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et commande ω (exprimée en rad/s) en fonction du temps (en secondes) avec la Stratégie 1 avec une distance d'établissement de 0.5 mètre (colonne de gauche) et de 5 mètres (colonne de droite).

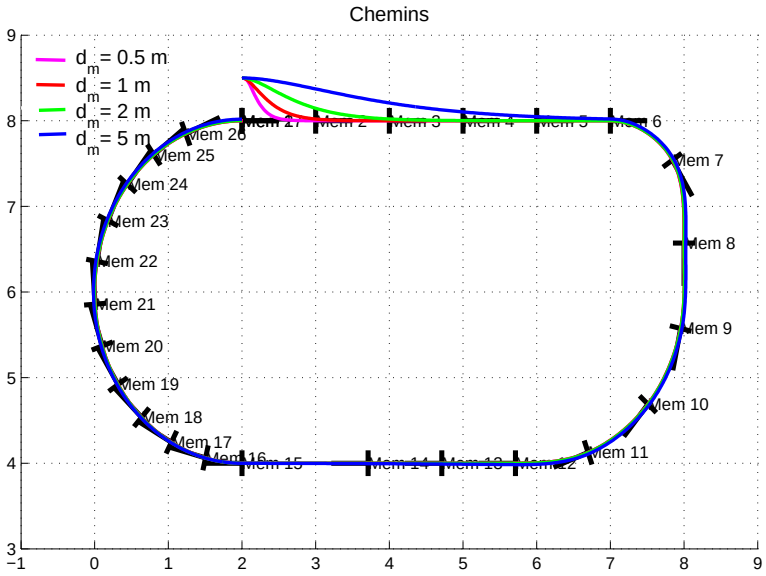


FIG. 3.17 – Chemins (représentés en coordonnés métriques) parcourus avec la Stratégie 2 et différentes distances d'établissement.

3.3.2.1 Facteur d'échelle

Pour les deux stratégies de suivi proposées, nous supposons que l'erreur latérale y peut être obtenue avec un facteur d'échelle s de valeurs 0.3, 0.6, 1 et 5. Nous analysons l'influence de ce facteur sur la réalisation de la tâche de navigation.

Stratégie 1 Les chemins parcourus sont représentés Fig. 3.18 avec les différents facteurs d'échelle. Quand le facteur d'échelle est inférieur à 1 ($s = 0.3$ ou $s = 0.6$), l'influence de l'erreur angulaire devient prépondérante par rapport à l'erreur latérale. Le chemin suivi dans les lignes droites est satisfaisant mais il se situe à l'intérieur des virages (l'erreur latérale n'est alors pas régulée à zéro). Lorsque l'échelle est largement surestimée ($s = 5$), des oscillations apparaissent. L'écart maximal observé avec le chemin appris est alors de 31 centimètres.

Stratégie 2 Les chemins parcourus pour ces différents facteurs d'échelle sont représentés Fig. 3.19. Les écarts entre les chemins suivis et le chemin appris ont pour moyenne 11 centimètres ($s = 0.3$), 2 centimètres ($s = 0.6$) et 0 centimètre ($s = 1$ et $s = 5$) et atteignent des valeurs maximales de 25 centimètres, 9 centimètres, 2 centimètres et 1 centimètres. Comme pour la Stratégie 1, une échelle sous-estimée conduit à un chemin parcouru situé à l'intérieur du virage. Si le facteur est supérieur ou égal à 1, les écarts entre les chemins appris et parcourus sont faibles. Contrairement à la Stratégie 1, l'écart latéral n'oscille pas lorsque le facteur d'échelle est surestimé.

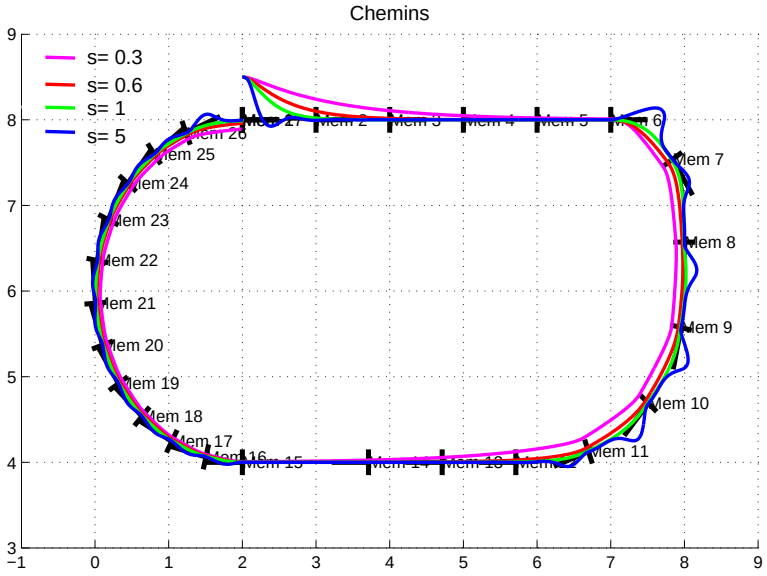


FIG. 3.18 – Chemins (représentés en coordonnées métriques et avec différentes couleurs) parcourus avec la Stratégie 1 et différents facteurs d'échelle.

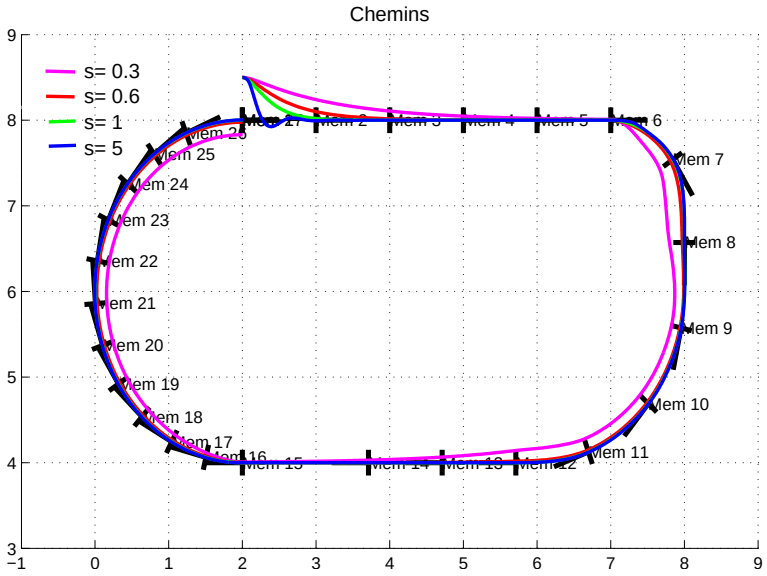


FIG. 3.19 – Chemins (représentés en coordonnées métriques avec différentes couleurs) parcourus avec une Stratégie 2 et différents facteurs d'échelle.

Dans le cadre de la Stratégie 1, l'échelle doit être correctement estimée ou légèrement surestimée. Il peut être nécessaire pour cela d'utiliser un autre capteur. La Stratégie 2 est ici plus adaptée si le facteur d'échelle est mal estimé, une surestimation de l'échelle étant suffisante pour obtenir un comportement satisfaisant.

3.3.2.2 Erreurs sur la pose du capteur

Jusqu'à maintenant, nous avons supposé que la pose du capteur était parfaitement connue dans le repère robot. On fixe la translation à ${}^{r}\mathbf{t}_c = [0.5\ 0\ 0.1]^\top$ avec une orientation nulle ${}^{r}\mathbf{R}_c = \mathbf{I}_{3\times3}$. Nous supposons que le facteur d'échelle est parfaitement connu. Nous étudions l'influence d'erreurs sur la pose du capteur dans le repère du robot. Soient $\widetilde{{}^{r}\mathbf{t}_c} = [\Delta X\ \Delta Y\ 0]^\top$ l'erreur sur la position et $\Delta\theta$ l'erreur sur la rotation autour de l'axe Z_c ajoutées à la pose et à l'orientation réelles.

Erreur de translation le long de l'axe d'avance On suppose que les erreurs ΔY et $\Delta\theta$ sont nulles. Nous étudions l'influence de trois valeurs pour l'erreur de position sur l'axe X_c :

- ΔX = -0.3 mètre,
- sans erreur ($\Delta X = 0$),
- $\Delta X = 0.3$ mètre.

Les résultats avec la Stratégie 1 sont représentés Figure 3.20. Le chemin sensoriel est entièrement suivi par le robot. On observe Fig. 3.20 (a) que si la valeur exacte de ΔX n'est pas connue, les écarts entre le chemin appris et les chemins parcourus sont plus importants que lorsque $\Delta X = 0$ (4 centimètres contre 1 centimètre en moyenne et avec des valeurs maximales de 13 centimètres contre 3 centimètres). En effet, la valeur de ΔX influence le changement d'image de référence. Dans le cas ΔX = -0.3 mètre, l'écart latéral est régulé à zéro tandis que l'écart angulaire dans les virages ne convergent que jusqu'à des valeurs d'environ 4 degrés (voir Fig. 3.20 (b)). Pour une erreur de position $\Delta X = 0.3$ mètre, des dépassements sont observés sur les variables d'état avant les changements d'image clé. Des résultats similaires sont obtenus avec la Stratégie 2.

Erreur de translation le long de l'axe latéral On suppose maintenant que les erreurs ΔX et $\Delta\theta$ sont nulles. Nous étudions l'influence de trois valeurs pour l'erreur de position sur l'axe Y_c :

- ΔY = -0.5 mètre,
- sans erreur ($\Delta Y = 0$),
- $\Delta Y = 0.5$ mètre.

Les résultats avec la Stratégie 1 sont représentés Figure 3.21. L'écart entre les chemins parcourus et le chemin appris est faible (les écarts moyens sont inférieurs à 2 centimètres avec des valeurs maximales de 8 centimètres, 2 centimètres et 7 centimètres). La régulation de l'écart angulaire est, cependant, perturbée par les erreurs de positions (voir

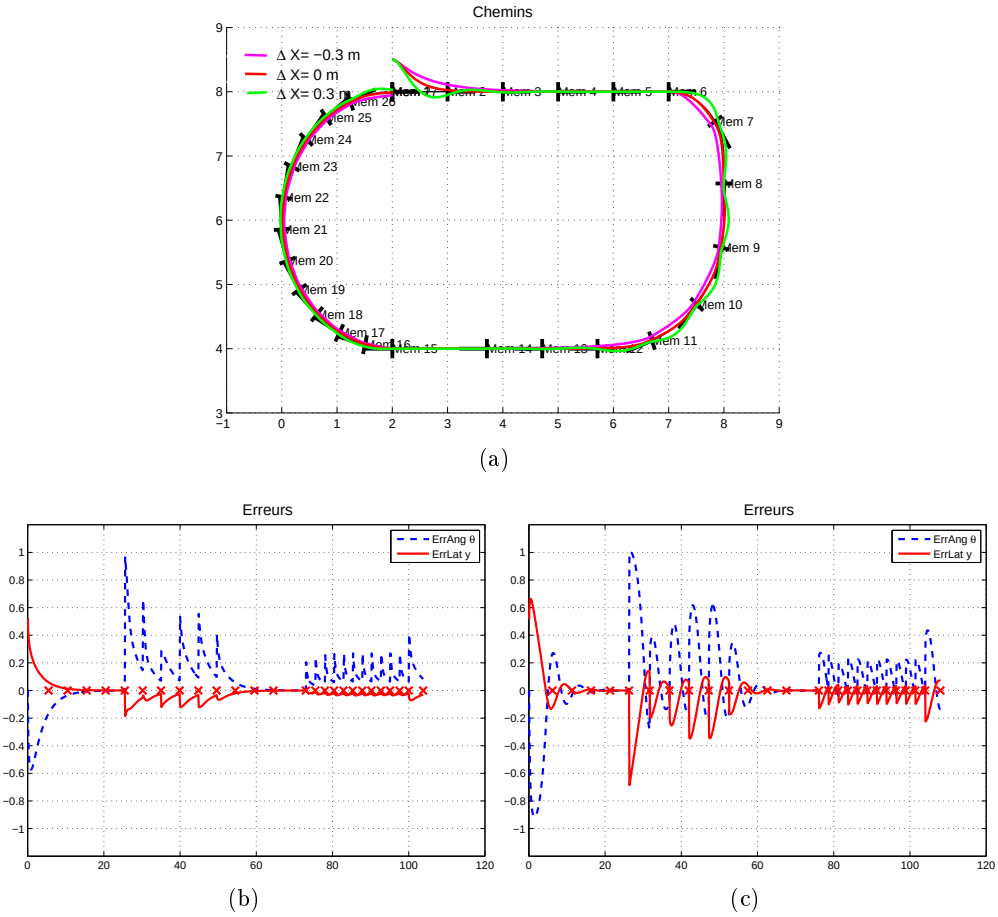
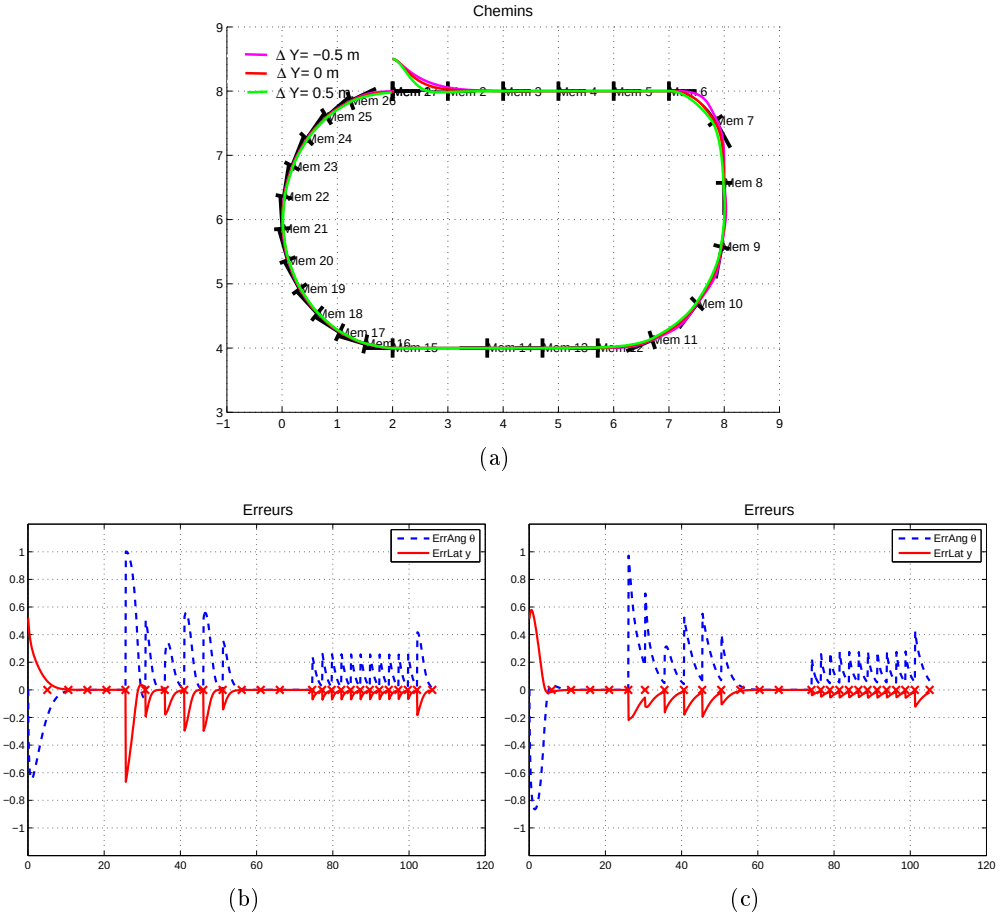


FIG. 3.20 – (a) chemins (représentés en coordonnées métriques) parcourus avec la Stratégie 1 et différentes erreurs de position de la caméra sur l'axe d'avance. Erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians en fonction du temps (en secondes) pour des erreurs de position (b) ΔX = -0.3 mètre et (c) $\Delta X = 0.3$ mètre.



(a)

(b) (c)

FIG. 3.21 – (a) chemins (représentés en coordonnées métriques) parcourus avec la Stratégie 1 et différentes erreurs de position de la caméra sur l'axe latéral. Erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians en fonction du temps (en secondes) pour des erreurs de position (b) ΔY = -0.5 mètre et (c) ΔY = 0.5 mètre.

Fig. 3.21 (c)). Des résultats similaires sont obtenus avec la Stratégie 2.

Erreur d'orientation autour de l'axe de rotation du véhicule On suppose maintenant que les erreurs ΔX et ΔY sont nulles. Nous introduisons les erreurs d'orientation suivantes :

- $\Delta\theta$ = -20 degrés,
- $\Delta\theta$ = -10 degrés,
- sans erreur ($\Delta\theta$ = 0 degré),
- $\Delta\theta$ = 10 degrés,
- $\Delta\theta$ = 20 degrés.

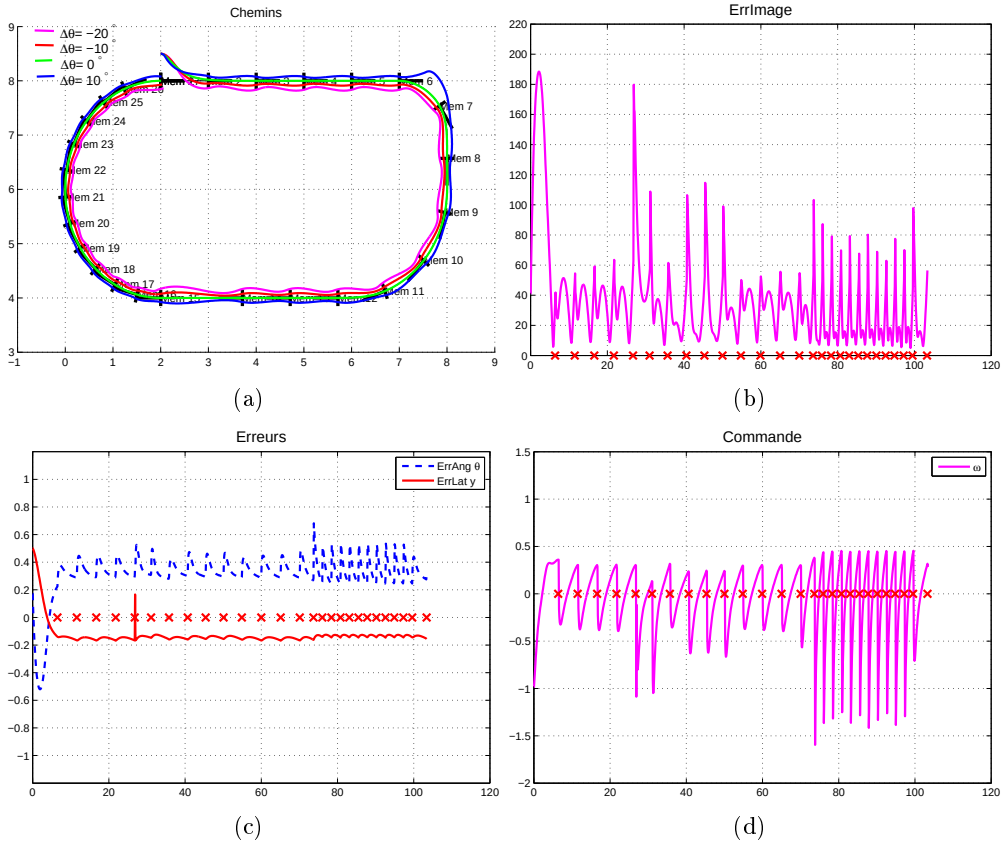


FIG. 3.22 – (a) chemins (représentés en coordonnées métriques) parcourus avec une Stratégie 2 et différentes erreurs d'orientation de la caméra. Erreur dans l'image (en pixels), erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et commande ω (exprimée en rad/s) en fonction du temps (en secondes) avec la Stratégie 2 et une erreur d'orientation du capteur de $\Delta\theta = -20$ degrés.

Les résultats avec la Stratégie 2 sont représentés Fig. 3.22, le chemin parcouru avec une erreur d'orientation $\Delta\theta = 20$ degrés n'étant pas représenté car la tâche de navigation échoue. L'influence d'une erreur sur ce paramètre est très importante comme on peut le constater Figure 3.22 (a). Les chemins sont parcourus avec des écarts de moyenne : 14 centimètres ($\Delta\theta$ = -20 degrés), 7 centimètres ($\Delta\theta$ = -10 degrés), nul ($\Delta\theta = 0$ degré) et 8 centimètres ($\Delta\theta = 10$ degrés) et de valeur maximale : 20 centimètres, 10 centimètres, 2 centimètres et 35 centimètres. De plus, des erreurs d'orientation importantes entraînent des oscillations. Les erreurs latérale et angulaire convergent vers des valeurs de -15 centimètres (resp. -14 centimètres) et de 16 degrés (resp. 14 degrés) sur le début du trajet (resp. sur la fin du trajet, lorsque les situations d'acquisition des images clés sont plus proches). Il est à noter qu'avec la Stratégie 1, les erreurs de suivi sont encore plus importantes.

Comme nous venons de le voir, les chemins parcourus sont satisfaisants en présence d'erreurs sur la position du capteur. De légères oscillations peuvent, cependant, apparaître en présence d'une erreur longitudinale. Cela est du à l'utilisation du Critère 2 de changement d'image clé qui dépend entre autres de l'abscisse curviligne du point de (Υ) le plus proche du centre du robot. Ce dernier étant mal estimé dans cette situation, des changements d'image de référence sont réalisés trop tôt ou trop tard. Ce problème peut probablement être résolu par l'utilisation d'un critère de changement d'image clé différent. Les simulations montrent également que l'orientation du capteur par rapport au repère de commande doit être correctement estimée afin d'assurer la réussite de la navigation.

3.3.3 Influence des paramètres relatifs à la perception

Nous supposons que les paramètres extrinsèques de la caméra ainsi que le facteur d'échelle sont parfaitement connus. Aux paramètres d'étalonnage ont été ajoutés un bruit additif d'une valeur de 20% sur les focales réelles, de 2 pixels sur les coordonnées exactes du point principal et de 20% sur le paramètre ξ. D'autre part, les erreurs de mesure sont représentées via un bruit aléatoire de distribution uniforme et de variance 0.5 pixels sur la position des points dans l'image. Les résultats sont représentés Fig. 3.23. Les chemins parcourus, non représentés ici, sont suivis avec des écarts de moyenne inférieure à 1 centimètre et de valeur maximale inférieure à 3 centimètres malgré les erreurs importantes sur les paramètres internes du capteur et sur les bruits de mesure. Les erreurs latérale et angulaire sont régulées à zéro (voir Fig. 3.23 (c) et (d)).

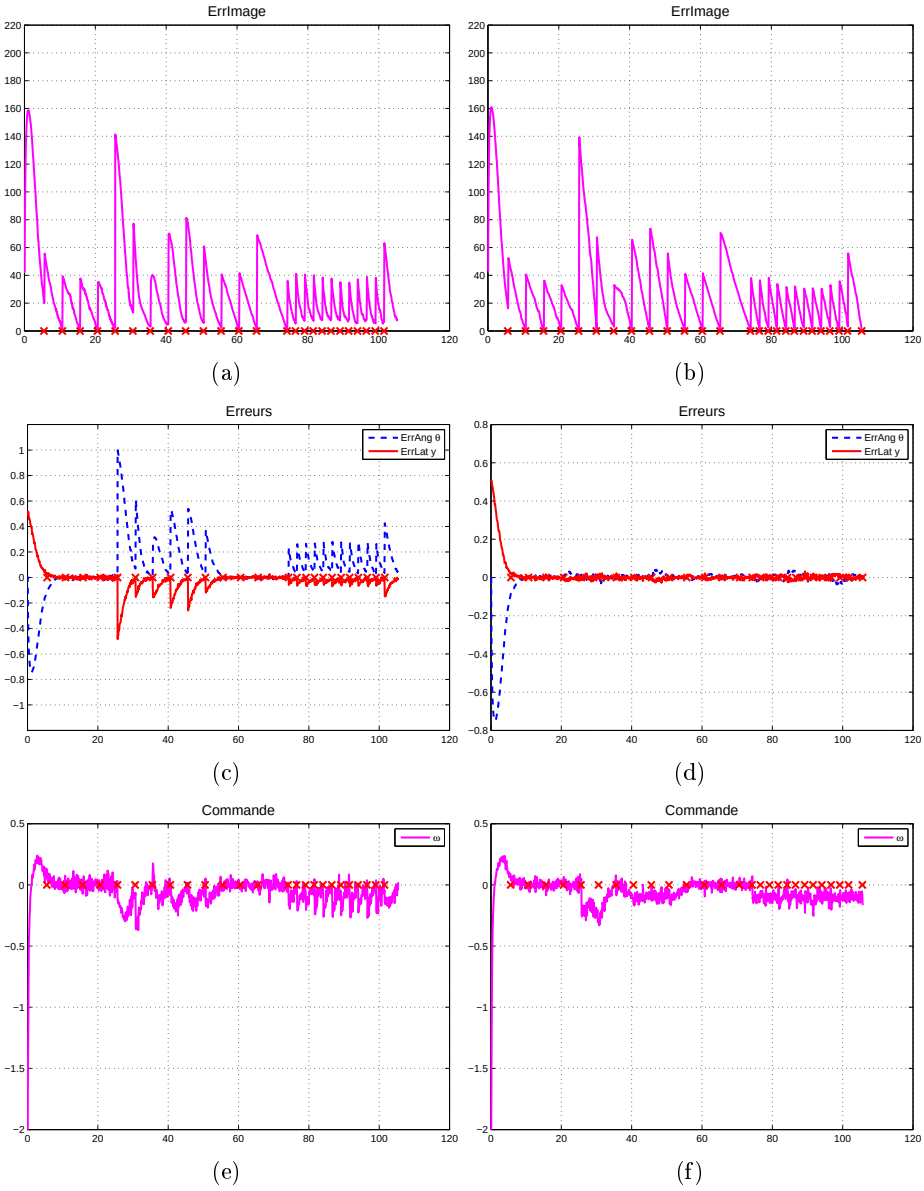


FIG. 3.23 – Erreur dans l'image (en pixels), erreur latérale y exprimée en mètre et erreur angulaire θ exprimée en radians et commande ω (exprimée en rad/s) en fonction du temps (en secondes) avec la Stratégie 1 (colonne de gauche) et la Stratégie 2 (colonne de droite) avec des erreurs d'étalonnage et des erreurs sur la perception (20% sur les focales, 2 pixels sur les coordonnées du point principal, 20% sur ξ et un bruit aléatoire de distribution uniforme et de variance 0.5 pixels sur les positions des points).

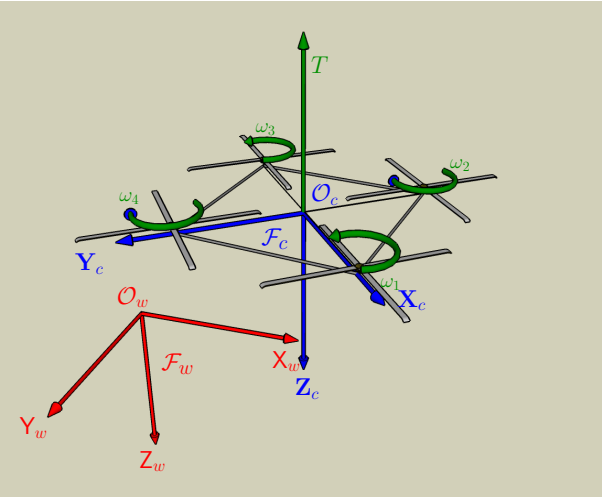


FIG. 3.24 – Repères utilisés pour modéliser le drone et position des 4 rotors générant la poussée T.

3.4 Cas des drones quadrirotors

Nous nous intéressons maintenant au suivi de chemin sensoriel pour les drones quadrirotors. Nous présentons tout d'abord quelques notations utiles à la bonne compréhension de la suite. Le modèle simplifié du quadrirotor que nous employons pour la définition de la commande est présenté dans la Section 3.4.2. Nous précisons ensuite l'objectif de commande pour ce type d'engin. Nous présentons finalement la loi de commande permettant de réaliser cet objectif dans la Section 3.4.4.

3.4.1 Notations

Deux repères seront utilisés dans la suite (voir Fig. 3.24) :

- un repère inertiel $\mathcal{F}_w(\mathcal{O}_w, \mathsf{X}_w, \mathsf{Y}_w, \mathsf{Z}_w)$ considéré Galiléen, lié à la terre, relatif à une origine fixe $\mathcal{O}_w$. Les axes de ce trièdre, considérés comme fixes, pointent vers le Nord (X_w), vers l'Est (Y_w) et vers le centre de la Terre (Z_w).
- un repère $\mathcal{F}_c(\mathcal{O}_c, \mathsf{X}_c, \mathsf{Y}_c, \mathsf{Z}_c)$ lié au corps du drone, ayant pour origine le centre de gravité $\mathcal{O}_c$ du véhicule. X_c est appelé axe de roulis et est dirigé vers l'avant du véhicule, Y_c est l'axe de tangage (dirigé vers la droite) et Z_c est l'axe de lacet, dirigé vers le bas.

L'orientation du repère lié au corps, exprimée dans le repère inertiel $\mathcal{F}_w$, peut être représentée de différentes manières (angles d'Euler, matrice de rotation, quaternions ...). Les angles RTL (Roulis, Tangage, Lacet) ou angles de Tait-Bryan, souvent employés dans le domaine aéronautique, permettent d'exprimer la relation de passage du repère $\mathcal{F}_c$ au repère $\mathcal{F}_w$ à partir d'une succession de trois rotations :

- une rotation d'angle ψ ($\psi \in]-180°,\ 180°]$), appelé angle de lacet, autour de Z_c

transformant $\mathcal{F}_c$ en un repère $\mathcal{F}_c^{'}$ défini par le trièdre $(\mathsf{X}_c^{'},\ \mathsf{Y}_c^{'},\ \mathsf{Z}_c^{'})$
- une rotation d'angle θ ($\theta \in [-90°,\ 90°[$), appelé angle de tangage, autour de $\mathsf{Y}_c^{'}$ transformant $\mathcal{F}_c^{'}$ en un repère $\mathcal{F}_c^{''}$ défini par le trièdre $(\mathsf{X}_c^{''},\ \mathsf{Y}_c^{''},\ \mathsf{Z}_c^{''})$
- une rotation d'angle ϕ ($\phi \in [-180°,\ 180°]$), appelé angle de roulis, autour de $\mathsf{X}_c^{''}$ transformant $\mathcal{F}_c^{''}$ en $\mathcal{F}_w$

Le vecteur $\xi = [\phi\ \theta\ \psi]^\top$ constitué respectivement des angles de roulis, tangage et lacet permet de représenter la rotation assurant le passage de $\mathcal{F}_c$ vers $\mathcal{F}_w$ ou *orientation* du drone. L'attitude du drone (également appelée *assiette*) est la rotation définie par les angles de roulis et de tangage.

Soit $\mathcal{SO}(3)$ le groupe des matrices de rotations de dimension 3 tel que :

$$\mathcal{SO}(3) = \left\{ \Theta \in \Re^{3\times 3} \mid \Theta^\top \Theta = \mathbf{I}_{3\times 3} \text{ et } det(\Theta) = 1 \right\}$$

La matrice de rotation $\Theta \in \mathcal{SO}(3)$ du drone associée aux angles RTL est de la forme :

$$\Theta = \begin{pmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi - s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta secondes\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi + s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix} \tag{3.22}$$

avec les notations : $s_\theta = \sin(\theta)$ et $c_\theta = \cos(\theta)$.

Soit Ω le vecteur instantané de rotation du drone exprimant la vitesse de rotation dans le repère $\mathcal{F}_c$. La dynamique de Θ est donnée en fonction de la vitesse de rotation Ω par la relation suivante :

$$\dot{\Theta} = \Theta\, [\Omega]_\times$$

où $[\Omega]_\times$ représente la matrice antisymétrique associée au vecteur Ω.

La position (respectivement la vitesse linéaire) du centre de gravité du robot exprimée dans le repère inertiel $\mathcal{F}_w$ est noté p (resp. v).

Le drone est actionné par 4 rotors disposés en croix (voir Fig. 3.24). En considérant que la dynamique des moteurs est négligeable par rapport à celle du drone, les vitesses de rotation des quatre rotors sont considérées atteintes immédiatement. La configuration du drone peut alors être décrite sans ambiguïté par le vecteur d'état : $\left[\mathsf{p}^\top\ \mathsf{v}^\top\ \xi^\top\ \Omega^\top\right]^\top$.

3.4.2 Modélisation des quadrirotors

Dans la littérature, deux méthodes sont principalement employées pour décrire la dynamique du drone : la première est basée sur le formalisme de Lagrange (par exemple dans [Castillo 04]) alors que la seconde utilise les équations de Newton (voir, par exemple, [Hamel 02b]). Nous utilisons plutôt la seconde méthode qui est, de notre point de vue, plus facilement interprétable. On pourra, par exemple, se référer à la thèse [Guénard 06] pour une modélisation complète des drones X4 via les équations de Newton.

Le bilan des forces impliquées dans cette modélisation fait intervenir : le poids du drone, le couple gyroscopique (résultant de la rotation simultanée de la structure du drone et de la rotation à grande vitesse des hélices), les forces de traînée, le couple de réaction (créé par le changement des vitesses de rotation des rotors) et T la poussée générée par les 4 moteurs. Dans les différents modèles dynamiques proposés dans la littérature ([Pounds 02, Bouabdallah 05, Waslander 05, Metni 05] entre autres), certaines de ces forces sont négligées.

Nous utiliserons également un modèle simplifié du quadrirotor pour la synthèse de la loi de commande. Nous supposons que la géométrie du drone est parfaite. Soient m la masse du drone et g la constante de gravité. Le poids, constant, s'applique au centre de gravité du drone et est dirigée dans la direction constante Z_w :

$$\mathsf{P} = mg\mathsf{Z}_w$$

Le drone est supposé être en vol quasi-stationnaire ce qui se traduit par les conditions suivantes :

- la vitesse v reste modérée ($\|\mathsf{v}\| < v_{lim}$, $v_{lim} \in \Re^{+*}$).
- la poussée T reste assez proche de celle de la poussée à l'équilibre ($mg - \delta \leq T \leq mg + \delta$ avec $\delta \in \Re^{+*}$ tel que $0 < \delta \ll mg$).

La première condition implique que les efforts dus au déplacement en translation (traînée) peuvent être négligés. Comme la géométrie du drone est parfaite et comme les rotors, en nombre pair, évoluent autour de leur vitesse d'équilibre (seconde condition), l'effet gyroscopique provoqué par la rotation des rotors est imperceptible et sera donc négligé dans la suite. Nous négligeons également le couple de réaction.

La force de poussée est donnée par :

$$\mathsf{F}_p = -T\Theta\mathsf{Z}_w$$

On note Γ_1, Γ_2 et Γ_3 les couples de commande relatifs aux angles RTL. Le vecteur $[T\ \Gamma_1\ \Gamma_2\ \Gamma_3]^\top = [T\ \mathsf{\Gamma}^\top]^\top$ peut être considéré comme entrée de commande virtuelle. Il est lié au vecteur des vitesses ω_i des rotors par la relation inversible suivante :

$$\begin{pmatrix} T \\ \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \end{pmatrix} = \begin{pmatrix} b & b & b & b \\ 0 & -db & 0 & db \\ db & 0 & -db & 0 \\ \kappa & -\kappa & \kappa & -\kappa \end{pmatrix} \begin{pmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{pmatrix} \tag{3.23}$$

où b et κ sont des constantes positives qui dépendent de la densité de l'air, du pas et des caractéristiques géométriques des hélices et d est la distance entre l'axe d'un rotor et le centre de gravité du robot.

Le moment dynamique du drone en $\mathcal{O}_c$ par rapport au repère inertiel $\mathcal{F}_w$ vaut :

$$\mathbf{I}\left.\frac{d\Omega}{dt}\right|_{\mathcal{F}_w}$$

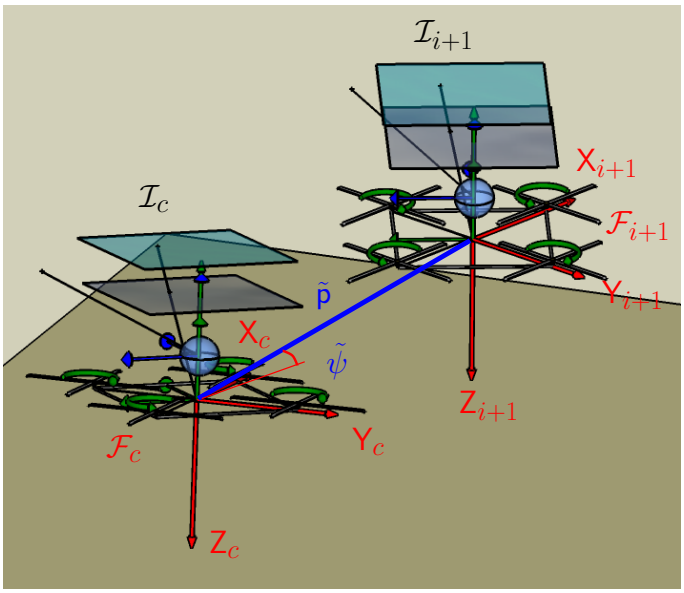


FIG. 3.25 – Suivi du chemin sensoriel.

où $\mathbf{I}$ est la matrice d'inertie du système exprimée au centre de gravité $\mathcal{O}_c$ dans le repère lié au corps $\mathcal{F}_c$. Comme la géométrie du drone est supposée parfaite, cette matrice est diagonale. Dans le repère $\mathcal{F}_c$, la dérivée de ce vecteur est définie comme :

$$\mathbf{I}\left.\frac{d\Omega}{dt}\right|_{\mathcal{F}_w} = \mathbf{I}\left.\frac{d\Omega}{dt}\right|_{\mathcal{F}_c} + \Omega \times \mathbf{I}\Omega = \mathbf{I}\dot{\Omega} + \Omega \times \mathbf{I}\Omega \tag{3.24}$$

Le modèle dynamique du drone peut alors s'écrire à partir des équations de Newton :

$$\left\{\begin{array}{rcl} \dot{\mathsf{p}} & = & \mathsf{v} \\ m\dot{\mathsf{v}} & = & -T\Theta\mathsf{Z}_w + mg\mathsf{Z}_w \\ \dot{\Theta} & = & \Theta\,[\Omega]_\times \\ \mathbf{I}\dot{\Omega} & = & -\Omega \times \mathbf{I}\Omega + \Gamma \end{array}\right. \tag{3.25}$$

3.4.3 Objectif de commande

Nous supposons que les paramètres extrinsèques du capteur (c'est-à-dire la position et l'orientation du repère lié au capteur exprimées dans le repère de commande du robot) sont supposés parfaitement connus. Soient $\mathcal{I}_i$ et $\mathcal{I}_{i+1}$ deux images consécutives du chemin sensoriel Ψ et $\mathcal{I}_c$ l'image courante. On notera $\mathcal{F}_i = (\mathcal{O}_i, \mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i)$, $\mathcal{F}_{i+1} = (\mathcal{O}_{i+1}, \mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})$ et $\mathcal{F}_c = (\mathcal{O}_c, \mathsf{X}_c, \mathsf{Y}_c, \mathsf{Z}_c)$ les repères associés au véhicule lors des prises de vue $\mathcal{I}_i$, $\mathcal{I}_{i+1}$ et $\mathcal{I}_c$ (voir Fig. 3.25).

En théorie, l'objectif de commande est identique à celui proposé pour les robots mobiles dans le paragraphe 3.2.1 :
L'objectif de commande est d'amener successivement le capteur, rigidement fixé au corps du robot, aux positions et aux orientations correspondant aux prises de vue définissant

le chemin sensoriel Ψ.

Cependant, les contraintes de déplacement du drone sont très différentes de celle des robots mobiles à roues. En particulier, la dynamique de rotation est couplée avec la dynamique de translation (voir Éq. (3.25)). Ainsi, le déplacement en translation est effectuée par une inclinaison en assiette du drone. Afin que la vitesse de translation du drone ne soit pas contrainte à être similaire à celle de la phase d'apprentissage au niveau des situations de référence, il est nécessaire de ne pas imposer l'inclinaison en assiette. Cela se traduit par le nouvel objectif de commande pour le quadrirotor :

L'objectif de commande est d'amener le capteur successivement aux positions et aux angles de lacet correspondant aux prises de vue définissant le chemin sensoriel Ψ.

Pour les robots à roues, nous avons employé une stratégie de suivi de chemin. Dans le contexte des drones, il n'existe, à notre connaissance, pas de travaux se rapportant à cette problématique. Nous proposons ici d'utiliser une stratégie de stabilisation pour réaliser cet objectif. Soient p et ψ (respectivement p_{i+1} et ψ_{i+1}) la position et le lacet du drone, exprimés dans le repère inertiel $\mathcal{F}_w$ et correspondant au repère $\mathcal{F}_c$ (resp. $\mathcal{F}_{i+1}$). L'objectif de commande se ramène alors à réguler la position p vers p_{i+1} et le lacet ψ vers ψ_{i+1}. On définit alors les erreurs de position $\tilde{\mathsf{p}} = \mathsf{p} - \mathsf{p}_{i+1}$ et de lacet $\tilde{\psi} = \psi - \psi_{i+1}$ (voir Fig. 3.25).

3.4.4 Commande

Notre objectif de commande peut être vu comme une tâche de positionnement et une tâche de régulation en lacet. Afin de synthétiser cette commande, nous proposons d'employer le système de commande hiérarchique représenté Fig. 3.26. Ce système a été choisi car sa stabilité a été démontrée dans [Guénard 08]. De plus, chaque niveau de commande peut permettre de compenser les perturbations internes à cette boucle. La commande en position consiste à assigner une vitesse de translation désirée v_d et à assurer que le système est en vol quasi-stationnaire.

Le système (3.25) étant sous-actionné et les dynamiques de rotation et de translation couplées, la commande en translation impose les consignes d'assiette. La loi de commande peut alors être dissociée en deux contrôleurs interdépendants permettant :

- le contrôle de la dynamique de translation, pour lequel la poussée et une consigne d'assiette sont définies,
- le contrôle de la dynamique de rotation, pour lequel les couples de commande sont définis.

La dynamique de translation est stabilisée par un régulateur embarqué comme celui proposé dans [Guénard 08]. Ce régulateur permet la convergence de la vitesse du centre de gravité v vers la vitesse désirée v_d, tout en garantissant que le véhicule reste en régime quasi-stationnaire, en imposant une assiette désirée et une poussée T. La rotation désirée Θ_d est obtenue en fusionnant l'assiette désirée avec le lacet désiré.

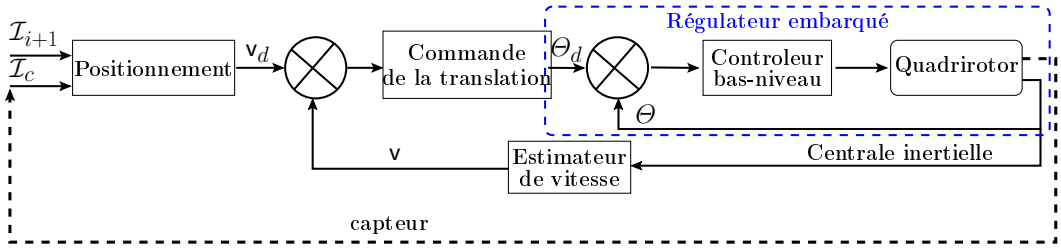


FIG. 3.26 – Schéma bloc de la commande hiérarchique du drone.

La dynamique de rotation est ensuite stabilisée par un régulateur qui permet la convergence de la matrice de rotation Θ vers la matrice de rotation désirée Θ_d en imposant les couples de commande Γ. Ce régulateur assure que l'assiette est limitée aux petits angles et que la vitesse de translation est bornée. Cela permet de rester dans les conditions de vol quasi-stationnaire si la vitesse désirée v_d permet cette condition.

La stabilité de ce régulateur embarqué a été démontrée dans [Guénard 06]. En pratique, le dispositif expérimental et le réglage des gains amènent une convergence rapide de l'orientation vers l'orientation désirée. Pour la dynamique de translation, les gains associés sont plus petits que les gains de la commande en rotation. De ce fait, en considérant la dynamique de v lente par rapport à celle de Θ, on peut considérer Θ_d comme lentement variable (quasiment constant). On peut ainsi négliger les termes de couplage entre les deux boucles imbriquées et assurer que Θ_d tend vers Θ et que v_d tend vers v.

Il nous reste maintenant à définir la vitesse désirée v_d permettant d'atteindre notre objectif de commande. Soient l'erreur de position $\tilde{\mathsf{p}}$ et l'erreur de vitesse de translation $\tilde{\mathsf{v}}$ définies par :

$$\left\{ \begin{array}{rcl} \tilde{\mathsf{p}} & = & \mathsf{p} - \mathsf{p}_{i+1} \\ \tilde{\mathsf{v}} & = & \mathsf{v} - \mathsf{v}_d \end{array} \right. \tag{3.26}$$

où p_{i+1} est la position désirée, supposée constante ($\dot{\mathsf{p}}_{i+1} = \mathsf{0}$).

Nous introduisons la fonction vectorielle $\mathrm{sat}_\epsilon(\mathsf{x})$ ($\epsilon \in \Re^{+*}$) représentant la saturation de chaque composante x_i du vecteur x à ϵ :

$$\left\{ \begin{array}{rcccl} \mathrm{sat}_\epsilon(x_i) & = & x_i & \text{si} & |x_i| \leq \epsilon \\ \mathrm{sat}_\epsilon(x_i) & = & \epsilon \ \mathrm{signe}(x_i) & \text{si} & |x_i| > \epsilon \end{array} \right.$$

Théorème 3.4.1 *La loi de commande suivante :*

$$\mathsf{v}_d = -K \mathrm{sat}_\epsilon(\tilde{\mathsf{p}}) \tag{3.27}$$

avec $K \in \Re^{+*}$ un scalaire petit comparé aux gains de la dynamique de translation et ϵ un scalaire qui dépend des conditions limites de vol quasi-stationnaire sur la vitesse de translation, permet de réguler l'erreur de position $\tilde{\mathsf{p}}$ à zéro.

Preuve: Afin de vérifier que cette loi de commande assure la stabilisation du drone, on considère la fonction de stockage suivante :

$$S = \frac{1}{2}||\tilde{\mathsf{p}}||^2 \tag{3.28}$$

En utilisant la première ligne du système (3.25) et la définition de l'erreur de position, la dérivée de S par rapport au temps peut s'écrire :

$$\dot{S} = \tilde{\mathsf{p}}^\top \mathsf{v} \tag{3.29}$$

Avec la loi de commande (3.27) et la définition de l'erreur de vitesse, l'équation (3.29) peut se réécrire sous la forme :

$$\dot{S} = -K\tilde{\mathsf{p}}^\top \mathrm{sat}_\epsilon(\tilde{\mathsf{p}}) + \tilde{\mathsf{p}}^\top \tilde{\mathsf{v}} \tag{3.30}$$

Le terme $\tilde{\mathsf{p}}^\top \tilde{\mathsf{v}}$ agit comme une perturbation sur la stabilisation de la position. Si le gain K est choisi petit comparé aux gains de la dynamique de translation, alors on peut considérer que v_d est lentement variable et que la dynamique de translation est plus rapide que celle de p. Dans ces conditions, on peut affirmer que $\tilde{\mathsf{v}}$ converge vers zéro. Le terme perturbateur converge alors rapidement vers zéro et l'équation (3.30) peut être réécrite :

$$\dot{S} \simeq -K\tilde{\mathsf{p}}^\top \mathrm{sat}_\epsilon(\tilde{\mathsf{p}})$$

Sachant que K est positif et que la fonction de saturation a pour propriété $\mathsf{x}^\top \mathrm{sat}_\epsilon(\mathsf{x}) > 0$ pour tout $\mathsf{x} \neq \mathsf{0}$, la relation $\dot{S} \leq 0$ est assurée. La fonction de stockage S est donc définie négative ce qui assure la convergence de p vers p_d. La loi de commande (3.27) est alors stabilisante pour le système (3.25) et assure que les conditions de vol quasi-stationnaire sont remplies.

Réalisation de nos objectifs La loi de commande (3.27) permet de réguler la position courante p vers la position p_{i+1} où l'image désirée a été acquise tandis que le lacet est régulé au niveau de la boucle de commande en orientation.

Notons finalement que lorsque l'erreur de translation est estimée à un facteur d'échelle près $s \in \Re^{+*}$ (ce qui est le cas avec une caméra), la loi de commande (3.27) est toujours stabilisante tant que Ks est petit comparé aux gains de la dynamique de translation.

3.5 Résultats de simulation

Nous proposons de réaliser des simulations sous Matlab afin d'analyser les performances de l'approche décrite précédemment. Le même environnement et la même caméra que pour les simulations avec les robots à roues (voir Section 3.3) sont employés. Les paramètres du modèle complet du drone correspondent aux paramètres d'un drone réel. En particulier, les commandes en orientation et en poussée sont réalisées toutes les 6 millisecondes (cadence du DSP embarqué sur le drone ; voir Section 5.2.2, p. 146). Des

bruits blancs sont ajoutés sur les estimations des angles de rotation et sur les vitesses de rotation et correspondent aux bruits obtenus, après filtrage des données de la centrale inertielle. D'autre part, la poussée n'est pas instantanée et les moteurs ont un temps de réponse de 13 millisecondes. Les couples de commande sont limités à 0.6 Nm. La fréquence d'acquisition des images est fixée à 10 Hz. Le repère caméra et le repère robot sont supposés confondus (l'axe optique de la caméra est donc dirigée vers le plafond). Le vecteur de translation pour passer du repère caméra au repère robot est donc nul : ${}^r\mathsf{t}_c = \mathsf{0}_{3\times1}$ et la matrice de rotation vaut l'identité : ${}^r\mathbf{R}_c = \mathsf{I}_{3\times3}$ (paramètres extrinsèques).

Le chemin parcouru lors de la phase d'apprentissage est, dans le plan $\mathsf{X}_w\mathsf{Y}_w$, similaire à celui utilisé pour les robots mobiles à roues dans la Section 3.3 et contient également des changements d'altitude (voir Fig. 3.27). Ainsi, la position sur Z_w est de 0 mètre lors de la première ligne droite. Cette position passe de 0 mètre à 1 mètre lors du virage 1 et reste stabilisée à cette hauteur. Enfin, lors du virage 3 la position sur Z_w passe de 1 mètre à 0 mètre. Afin d'être proche des conditions expérimentales réelles, l'angle de lacet est constant tout le long du trajet parcouru et est fixée à $\theta = 0$ deg.

Les positions du robot correspondant aux acquisitions des images de référence sont situées tous les 1 mètre au début du déplacement, puis tous les 0.5 mètre lors du dernier virage. La mémoire visuelle est donc formée de 27 images de référence. Nous considérons que lors de l'apprentissage, la vitesse de translation est très faible. Les angles d'assiette des lieux d'acquisition des images sont donc considérés comme nuls.

L'appariement des points entre deux images est réalisé en simulation et permet d'estimer la matrice d'homographie en utilisant l'équation (4.14) définie p. 126. Les variables d'état du robot sont estimées à partir de la décomposition de cette matrice comme nous le verrons dans le Chapitre 4. On obtient alors aisément l'erreur de lacet $\tilde{\psi}$ et l'erreur de position $\tilde{\mathsf{p}}$ (à un facteur d'échelle près). Dans la suite, on notera $\tilde{\mathsf{p}} = [ErrX\ ErrY\ ErrZ]^\top$. La saturation sur la position est fixée à $\epsilon = 0.5$ mètre.

Dans un premier temps, nous supposons que les paramètres extrinsèques et intrinsèques de la caméra sont parfaitement connus, que les données capteur sont non bruitées et que le facteur d'échelle est connu de manière exacte. Nous analysons alors l'influence des paramètres intervenant dans la commande (critère de changement d'image clé, saturation). Nous analysons ensuite l'influence d'erreurs sur les paramètres relatifs à l'estimation des variables d'état (facteur d'échelle et paramètres extrinsèques du capteur). Nous proposons ensuite d'analyser la robustesse de l'approche vis-à-vis d'erreurs sur les paramètres intrinsèques du capteur visuel et en présence de bruits de mesure. Enfin, l'influence de perturbations (biais sur l'estimation de la vitesse de translation, perturbation sur les couples, perturbations de type vent) est analysée.

3.5.1 Influence des paramètres intervenant dans la commande

Ces paramètres sont les critères de changement d'image clé, les erreurs initiales et la valeur de la saturation.

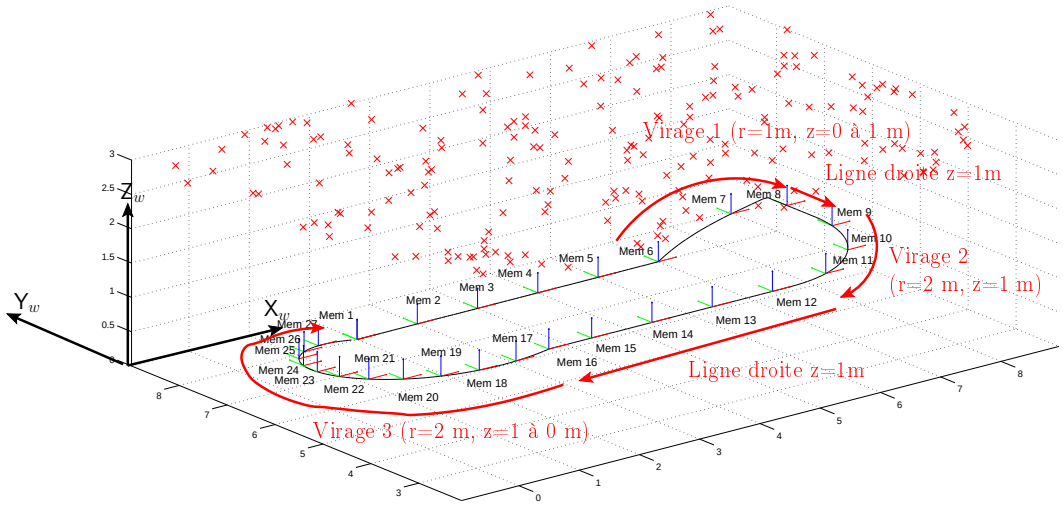


FIG. 3.27 – Environnement de simulation utilisé pour la navigation d'un quadrirotor. Les coordonnées sont exprimées en mètres. Les points 3D utilisés pour l'estimation des entrées de la commande sont représentés par des croix.

3.5.1.1 Influence du critère de changement d'image clé

Deux critères de changement d'image clé sont étudiés ici :

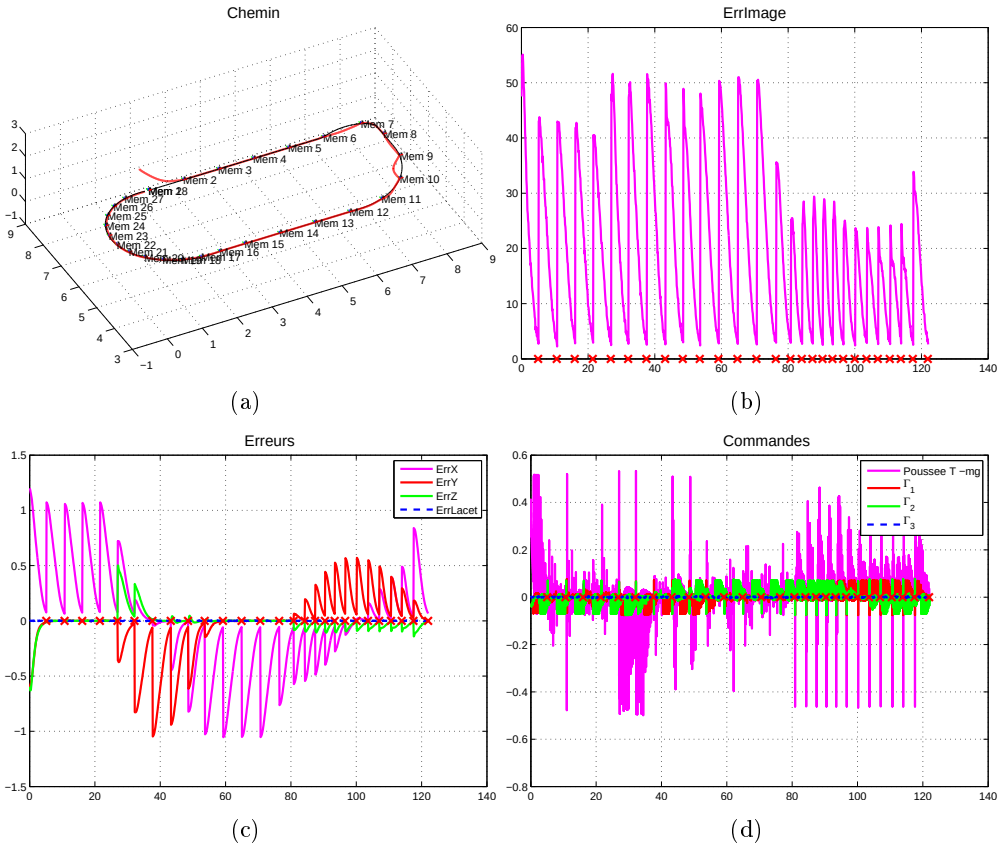
Critère 1 : le premier est défini dans l'espace capteur : une image clé est atteinte lorsque l'erreur dans l'image ErrImage est inférieure à un seuil.

Critère 2 : le second est défini dans l'espace des variables d'entrée de la commande : une image clé est atteinte lorsque la norme de l'erreur de position $\|\tilde{\mathbf{p}}\|$ est inférieure à un seuil.

Nous analysons l'influence du critère de changement d'image sur la réalisation de l'objectif de suivi. La position initiale du robot est $[2\ 8.5\ 0.5]^\top$ ($\tilde{\mathbf{p}} = [1\ 0.5\ \ -0.5]^\top$). L'erreur initiale dans l'image est d'environ 50 pixels. Nous considérons le Critère 1 avec un seuil fixé à 3 pixels. Les résultats sont représentés Figure 3.28. Lors de la réalisation de la tâche de navigation, l'écart entre le chemin parcouru et le chemin appris a pour moyenne 2 centimètres et écart type 3 centimètres avec un maximum de 19 centimètres [4]. Les erreurs en position et en lacet convergent vers zéro et les erreur dans l'image décroissent jusqu'au seuil fixé avant chaque changement d'image clé.

On considère maintenant le Critère 2 avec un seuil fixé à 0.02 mètre. Les résultats sont représentés Figure 3.29. On observe des résultats similaires à ceux obtenus précédemment.

[4] Les écarts ne sont pris en compte qu'à partir de l'image Mem2.



(a) (b)

(c) (d)

FIG. 3.28 – Critère 1 (ErrImage < 3 pixels) avec une erreur initiale $\tilde{\mathsf{p}} = [1\ -0.5\ -0.5]^\top$. (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels), (c) erreurs de position $\tilde{\mathsf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians et (d) commande (poussée T exprimée en Newton et couples Γ_1, Γ_2 et Γ_3 exprimés en Nm) en fonction du temps (en secondes).

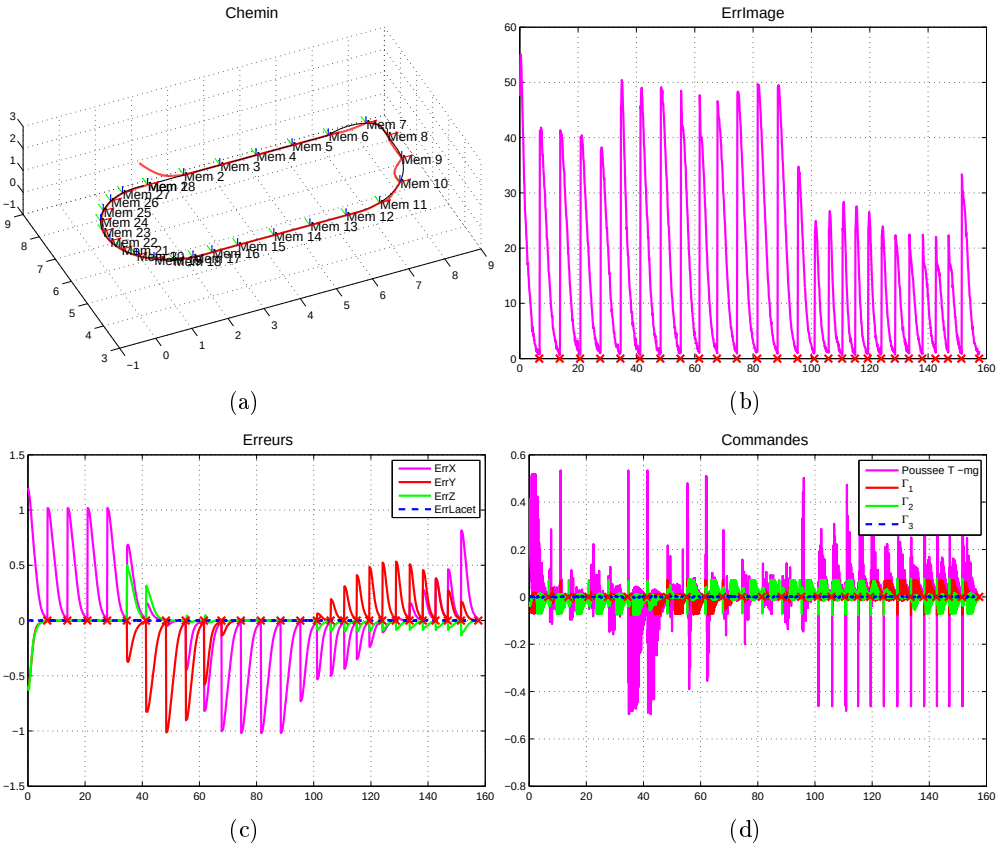


FIG. 3.29 – Critère de changement défini dans l'espace cartésien (erreur de position inférieure à 0.02 m) avec une erreur initiale $\tilde{\mathsf{p}} = [1 \ -0.5 \ -0.5]^\top$. (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels), (c) erreurs de position $\tilde{\mathsf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians et (d) commande (poussée T exprimée en Newton et couples Γ_1, Γ_2 et Γ_3 exprimés en Nm) en fonction du temps (en secondes).

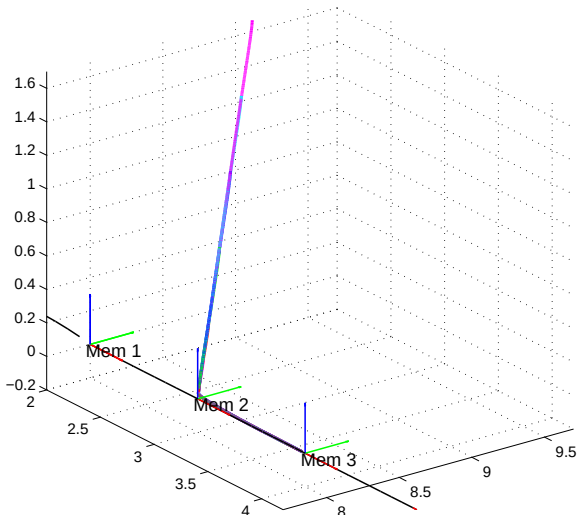


FIG. 3.30 – Chemins parcourus avec différentes positions initiales (coordonnées exprimées en mètres).

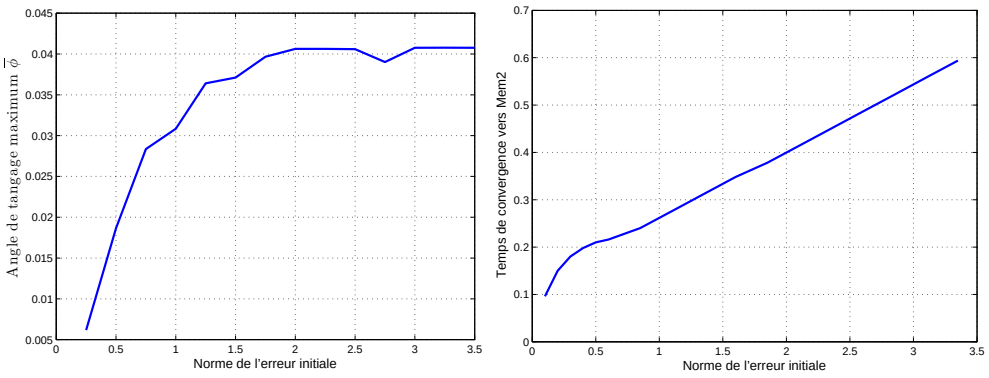
Les deux critères de changement d'image clé proposés sont tous les deux satisfaisants. Cependant, nous n'avons pas pris en compte le fait que les images pouvaient être de mauvaise qualité comme cela est le cas dans nos expérimentations (voir Chapitre 5). Cela résulte en des perturbations sur les erreurs de position et de lacet mais également sur l'erreur dans l'image. Dans ce cas, le Critère 2 de changement d'image clé nous semble plus approprié car il est plus robuste vis-à-vis de ces perturbations.

3.5.1.2 Norme de l'erreur de position

Nous analysons maintenant le comportement du drone en fonction de l'erreur de position. On rappelle que la saturation sur la position est fixée à $\epsilon = 0.5$ mètre. Nous considérons l'objectif de rejoindre l'image clé Mem2, le drone étant situé à la position $[3-\delta \quad 8+\delta \quad \delta]^\top$ avec δ telle que la norme de l'erreur initiale soit égale à 0.25 mètre, 0.5 mètre, 1 mètre, 1.5 mètre, 2 mètres puis 2.5 mètres. On rappelle que chaque composante de l'erreur de position est saturée par ϵ avant d'alimenter la loi de commande.

Les résultats sont représentés Figure 3.30. Pour ces différentes erreurs initiales, la position désirée est toujours atteinte. Il est à noter que les chemins parcourus sont similaires.

On note $\overline{\phi}$ la valeur absolue de l'angle de tangage maximum du drone observé sur la trajectoire pour rejoindre la position désirée. Sur la figure 3.31 (a), $\overline{\phi}$ est représentée en fonction de la norme initiale de l'erreur de position. Pour une distance supérieure à 1.5 mètre, $\overline{\phi}$ atteint une valeur de 0.04 radians (soit 2.8 degrés). On observe également que le temps de convergence vers la position désirée est approximativement linéaire en



(a) (b)

FIG. 3.31 – (a) Angle de tangage maximum $\overline{\phi}$ (en radians) et (b) temps de convergence pour rejoindre Mem2 (en secondes) en fonction de l'erreur initiale (exprimée en mètres) pour une valeur de saturation de l'erreur de position $\tilde{\mathsf{p}}$ valant $\epsilon = 0.5$ mètre.

fonction de la norme initiale pour une norme de l'erreur supérieure à 0.7 mètre (voir Fig. 3.31 (b)).

Les chemins parcourus avec différentes normes de l'erreur de position initiale sont similaires. La saturation limite les angles d'assiette maximum lorsque la valeur de la norme est importante, permettant ainsi de rester en vol quasi-stationnaire.

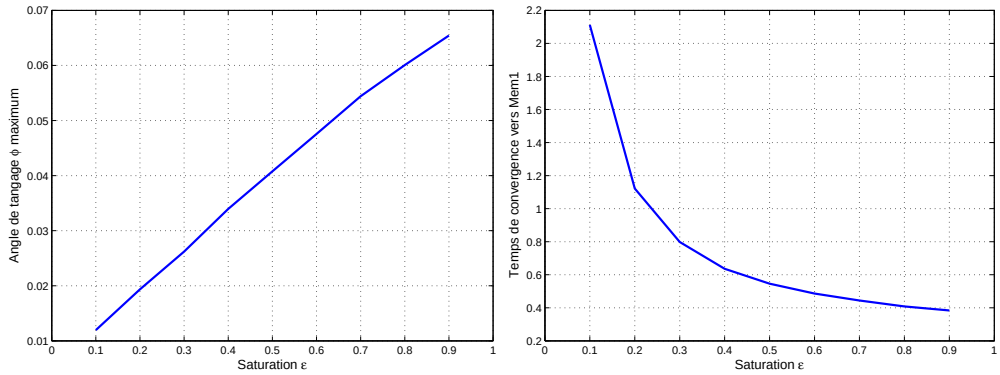
3.5.1.3 Influence de la saturation

Suite aux simulations précédentes, nous étudions l'influence de la valeur de la saturation ϵ. On fait varier cette saturation de 0.1 mètre à 0.9 mètre. Sur la Figure 3.32 (a), $\overline{\phi}$ et le temps de convergence pour une distance de 3 mètres sont représentés en fonction de la valeur de la saturation ϵ. Plus ϵ est grand, plus le temps nécessaire pour rejoindre l'image clé (avec une erreur initiale de 3 mètres) est faible (voir Fig. 3.32 (b)). Cependant, on observe que $\overline{\phi}$ est quasi proportionnelle à la saturation. Ainsi, l'angle de tangage maximum croît quasi linéairement avec la saturation. Cela conduit à une forte inclinaison du drone pouvant amener à un décrochage.

Dans la suite, on fixera la valeur de la saturation à $\epsilon = 0.5$ mètre ce qui permet un bon compromis entre rapidité et sécurité comme nous venons de le voir.

3.5.2 Influence des paramètres relatifs à l'estimation des variables d'état

Nous étudions maintenant l'influence des paramètres relatifs à l'estimation des variables d'état : le facteur d'échelle et la pose de la caméra (paramètres extrinsèques).



(a) (b)

FIG. 3.32 – (a) Angle de tangage maximum $\overline{\phi}$ (en radians) observé pour des normes d'erreur initiale supérieures à 1.5 mètre et (b) temps de convergence jusqu'à Mem2 (en secondes) en fonction de la valeur de la saturation ϵ (exprimée en mètres).

3.5.2.1 Facteur d'échelle

Le critère de changement d'image clé est défini dans l'espace cartésien et est fixé à $\|\tilde{\mathsf{p}}\| = 0.03$ mètre. Nous supposons que les erreurs de position sont données avec un facteur d'échelle s égal à 1/3, 1 et 3. Les résultats sont représentés Fig. 3.33.

Avec les différents facteurs d'échelle, le drone rejoint la situation cible en parcourant des chemins différents (voir Fig. 3.33). Les écarts par rapport au chemin appris ont pour moyenne 3 centimètres ($s = 1/3$), 2 centimètres ($s = 1$), 3 centimètres ($s = 5$) et pour valeur maximale 26 centimètres, 17 centimètres et 24 centimètres.

Le chemin parcouru est plus proche du chemin appris lorsque l'échelle est correctement estimée (on a alors un chemin parcouru de 21.5 mètres pour un chemin appris de 21.4 mètres). Avec un facteur $s = 1/3$, le chemin parcouru est plus court tandis qu'avec un facteur de 3, le chemin est plus long (26.5 mètres). En effet, un facteur d'échelle supérieur à 1 entraîne une surestimation des erreurs de translation ce qui conduit à des commandes d'amplitudes plus importantes (voir Fig. 3.34). On observe que les mouvements obtenus avec $s = 3$ sont de fortes amplitudes (voir Fig. 3.34, dernière ligne) ce qui peut être problématique pour la sécurité du matériel embarqué. En outre, les données acquises par les capteurs étant directement liées à ces mouvements, l'estimation des variables d'état avec l'image de la mémoire peut s'avérer difficile.

La tâche de navigation autonome a été réalisée correctement avec différents facteurs d'échelle. On peut, cependant, améliorer le comportement en vol du drone en estimant le facteur d'échelle via les capteurs embarqués sur ce type d'engin, notamment un télémètre à ultrason.

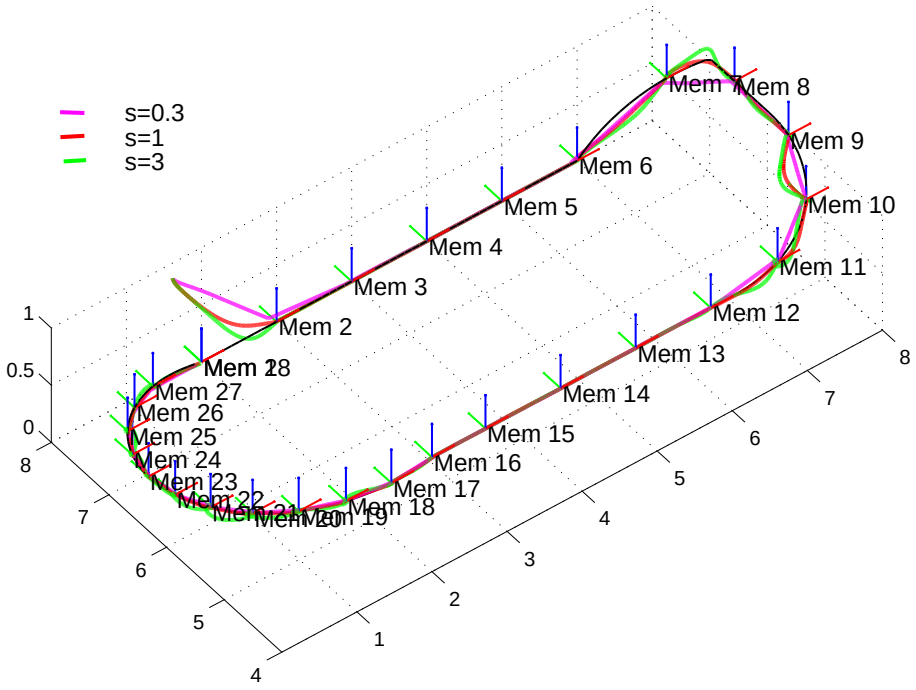


FIG. 3.33 – Influence du facteur d'échelle : chemins parcourus (coordonnées exprimées en mètres).

3.5.2.2 Erreur d'estimation de la pose du capteur

Jusqu'à maintenant, nous avons supposé que la pose du capteur était parfaitement connue dans le repère robot. On fixe cette pose à ${}^{r}\mathbf{t}_c = [0\ 0\ 0]^\top$ avec une orientation nulle ${}^{r}\mathbf{R}_c = \mathbf{I}_{3\times3}$. Nous supposons que le facteur d'échelle est parfaitement connu.

Erreur de position Soient $\widetilde{{}^{r}\mathbf{t}_c} = [\Delta X\ \Delta Y\ 0]^\top$ l'erreur sur la position ajoutées à la pose réelle du capteur dans le repère du robot. Nous étudions l'influence de cette erreur sur la tâche de navigation. Les résultats sont représentés Fig. 3.35 (a) pour une erreur de pose sur X_c de ΔX = -0.1 mètre, $\Delta X = 0$ mètre et $\Delta X = 0.1$ mètre, ΔY étant nulle puis Fig. 3.35 (b) pour une erreur de pose sur Y_c avec les mêmes valeurs, ΔX étant nulle.

Dans ces deux cas, on observe que les chemins parcourus lors de la phase autonome sont très proches de celui effectué lors de la phase d'apprentissage : la moyenne de l'erreur entre les chemins parcourus et le chemin appris est inférieure à 2 centimètres avec un écart type de 2 centimètres et une valeur maximale inférieur à 20 centimètres.

Erreur de lacet Soient $\Delta\psi$ l'erreur sur le lacet ajoutée à l'orientation en lacet réelle du capteur dans le repère du robot.

- $\Delta\psi$ = -20 degrés,
- $\Delta\psi$ = -10 degrés,

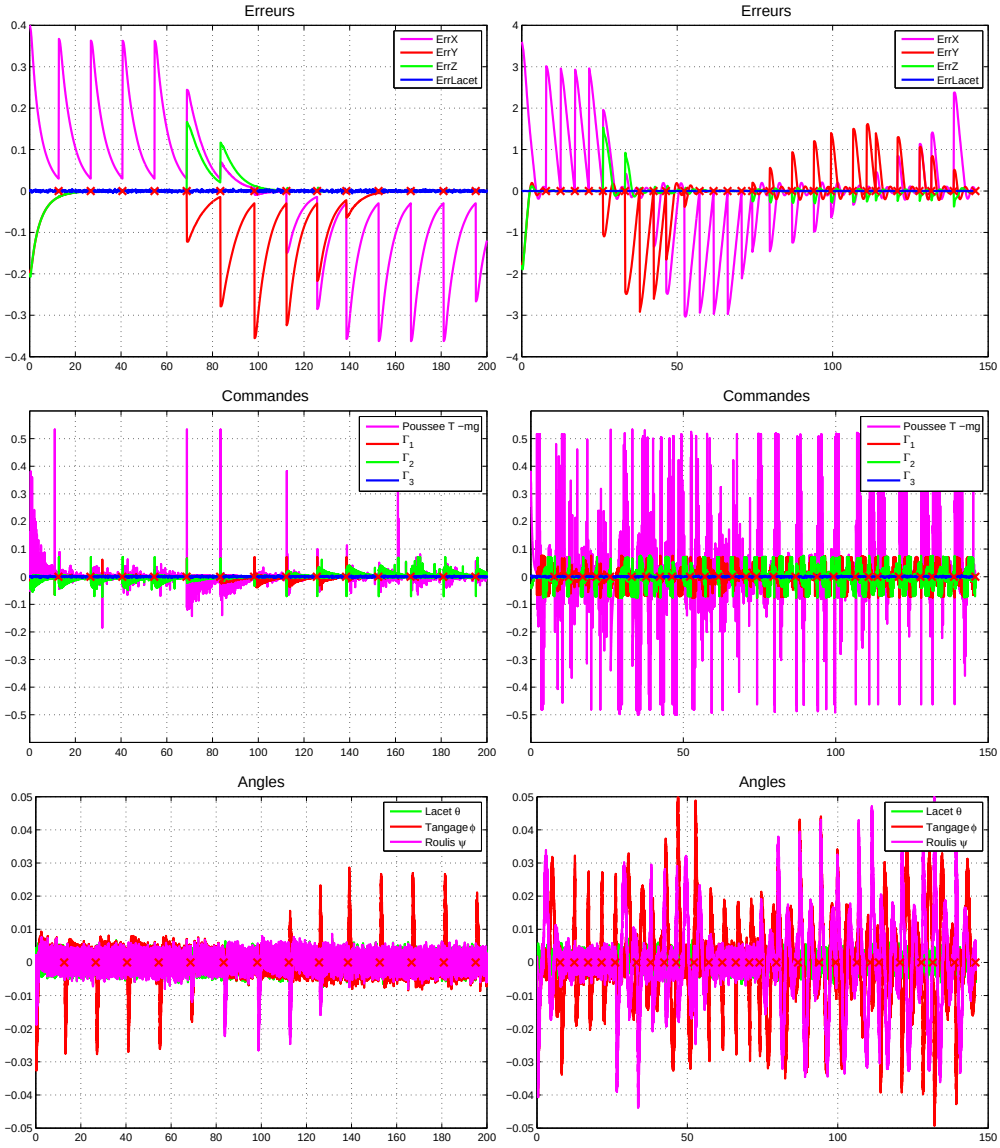
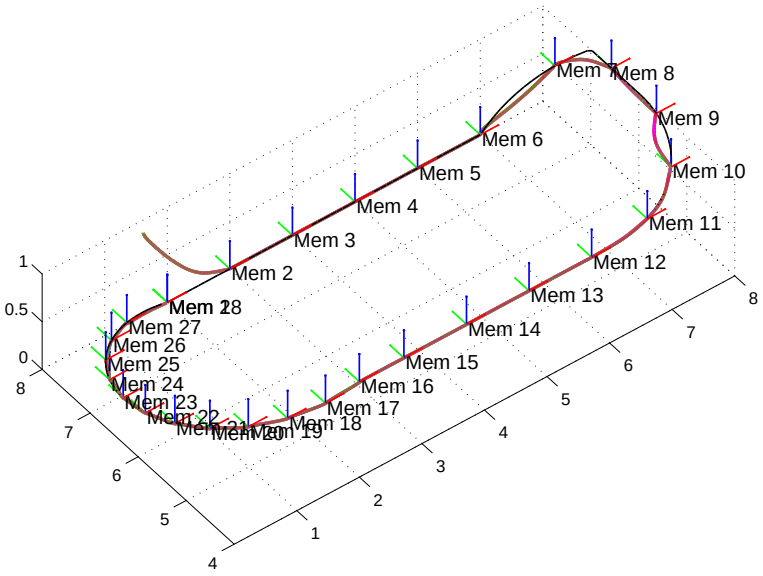
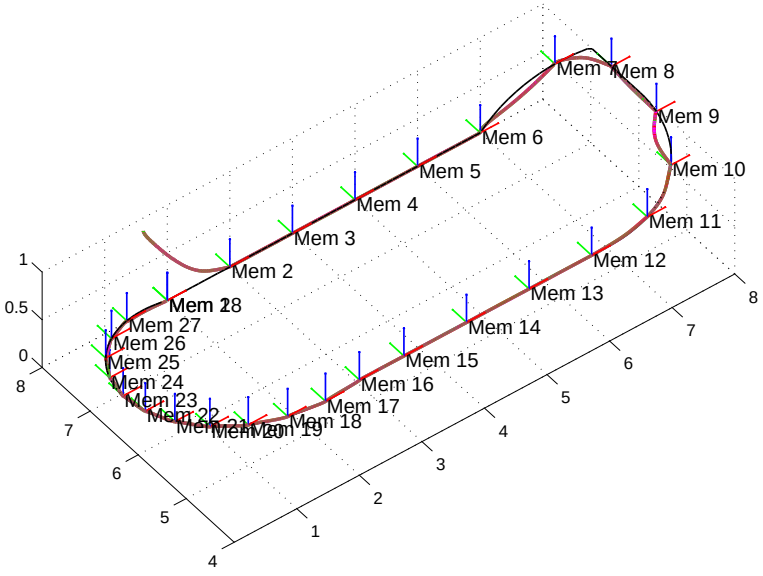


FIG. 3.34 – Influence du facteur d'échelle : erreurs de position $\tilde{\mathbf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians, commande (poussée T exprimée en Newton et couples Γ_1, Γ_2 et Γ_3 exprimés en Nm) et orientation du drone en fonction du temps (en secondes) avec un facteur d'échelle de 1/3 (colonne de gauche) et de 3 (colonne de droite)



(a) Erreur de position du capteur sur X_c



(b) Erreur de position du capteur sur Y_c

FIG. 3.35 – Influence d'une erreur de position de la caméra (a) sur X_c et (b) sur Y_c. Chemins parcourus (coordonnées exprimées en mètres) pour différentes valeurs de ces erreurs.

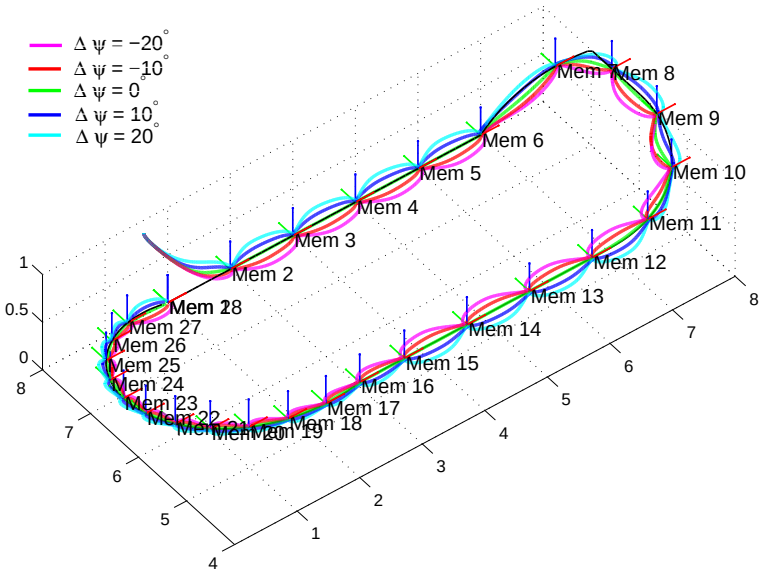


FIG. 3.36 – Influence d'une erreur de lacet de la caméra. Chemins parcourus (coordonnées exprimées en mètres) pour différentes valeurs de cette erreur.

- sans erreur ($\Delta\psi = 0$ degré),
- $\Delta\psi = 10$ degrés,
- $\Delta\psi = 20$ degrés.

Pour les différentes valeurs de l'erreur, le chemin sensoriel est entièrement suivi (voir Fig. 3.36).

À travers ces simulations, nous observons que la stratégie proposée est relativement robuste vis-à-vis des erreurs sur l'estimation des paramètres extrinsèques du capteur.

3.5.3 Influence des paramètres relatifs à la perception

Nous nous intéressons à l'influence des paramètres de la caméra sur la tâche de navigation. Aux paramètres d'étalonnage ont été ajoutés un bruit additif d'une valeur de 20% sur les focales réelles, de 2 pixels sur les coordonnées exactes du point principal et de 20% sur le paramètre ξ. D'autre part, les erreurs de mesure sont représentées par un bruit aléatoire de distribution uniforme et de variance 0.5 pixel sur la position des points dans l'image. Les résultats sont représentés Figure 3.37. Le chemin est suivi avec une erreur de moyenne de 5 centimètres, un écart type de 3 centimètres et une valeur maximale de 14 centimètres. L'erreur dans l'image décroît jusqu'à atteindre 3 pixels pour chaque image clé (Fig. 3.37 (b)). La stratégie adoptée est donc relativement robuste vis-à-vis de ce type d'erreurs.

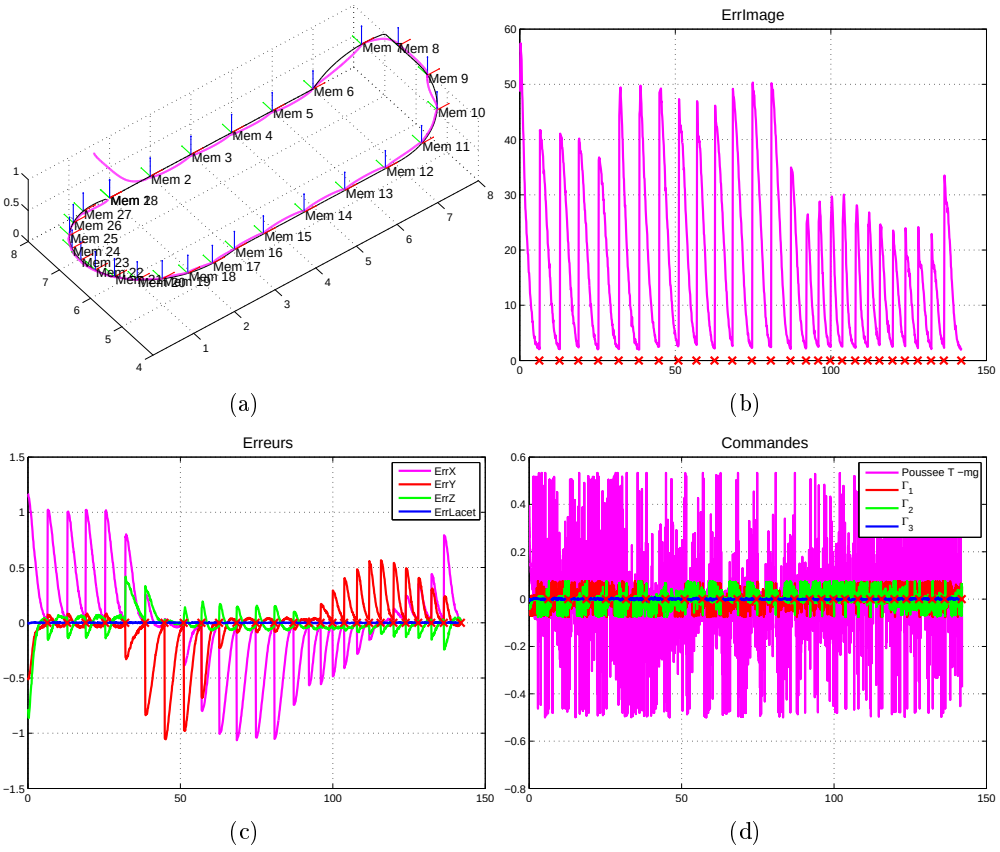


FIG. 3.37 – Influence des paramètres de perception. (a) chemin parcouru (coordonnées exprimées en mètres) et (b) erreur dans l'image (en pixels), (c) erreurs de position $\tilde{\mathbf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians et (d) commande (poussée T exprimée en Newton et couples Γ_1, Γ_2 et Γ_3 exprimés en Nm) en fonction du temps (en secondes).

3.5.4 Influence des perturbations

Nous supposons que les paramètres extrinsèques et intrinsèques de la caméra sont parfaitement connus, que les données capteur sont non bruitées et que le facteur d'échelle est connu de manière exacte. On considère ici des perturbations sur l'estimation de la vitesse de translation, sur les couples puis des perturbations de type vent.

3.5.4.1 Influence d'un biais sur la vitesse de translation

On considère un biais b fixe sur la mesure de la vitesse sur l'axe Y_c. Le critère de changement d'image clé est défini comme le changement de signe de la position sur X_c. Les résultats sont représentés Fig. 3.38 (a) avec un biais de 0 m/s, 0.025 m/s, 0.05 m/s, 0.1 m/s puis 0.2 m/s.

Les erreurs de position sur X_c et sur Z_c convergent vers 0 tandis que le biais sur la vitesse se traduit par une erreur statique de position sur Y_c, proportionnelle au biais sur la vitesse avec un rapport $\frac{1}{K}$ avec K le gain de la commande (voir Fig. 3.38 (b)).

Afin d'éviter des erreurs sur la position, il est nécessaire d'estimer la vitesse de translation sans biais. En général, les drones disposent d'une centrale inertielle mesurant les accélérations en translation. En théorie, la vitesse de translation peut être estimée par intégration des accélérations. Cependant, cette mesure dérive assez rapidement. D'autres capteurs doivent donc être employés. Nous verrons dans le Chapitre 5.2.2 que sur notre plateforme expérimentale, la vitesse de translation est estimée en utilisant les images acquises par une caméra embarquée dirigée vers le sol.

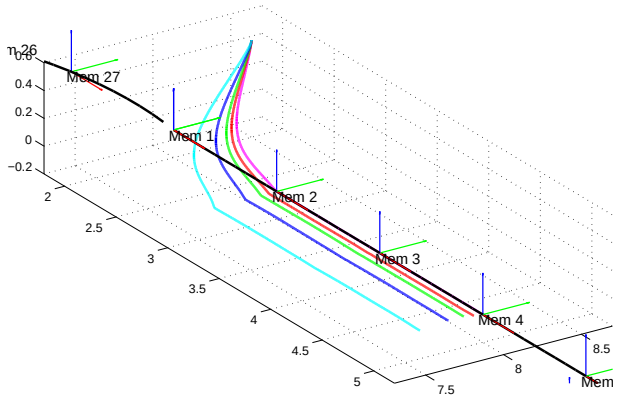
3.5.4.2 Perturbations sur les couples

On considère dorénavant que la vitesse de translation mesurée est correcte. L'erreur initiale vaut $\tilde{\mathsf{p}} = [1 \ -0.5 \ -0.5]^\top$. Des couples de perturbation de 0 à 0.2 rad/s ont été introduits sur l'axe X_c puis sur l'axe Y_c. Les résultats présentés Fig. 3.39 et Fig. 3.40 montrent une bonne robustesse vis-à-vis de ce type de perturbation. On observe sur la Fig. 3.40 (a) que les chemins suivis sont diffèrents en présence d'erreurs sur le couple Γ_2. Toutefois, les erreurs en position et en lacet convergent vers $\mathsf{0}$.

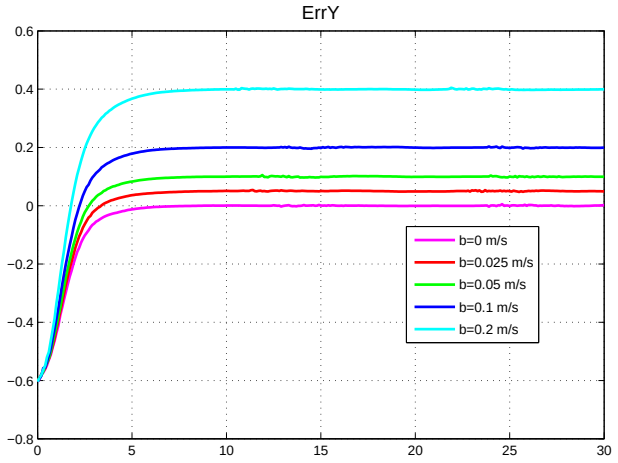
On note ici l'intérêt d'un schéma de commande avec des boucles imbriquées : la boucle de commande en assiette stabilise l'assiette même en présence de perturbations extérieures sur les couples.

3.5.4.3 Influence du vent

On considère la vitesse de translation comme connue de manière fiable et les perturbations autres que le vent nulles. Une perturbation de valeur fixe, due au vent, est ajoutée dans la direction X_w. Les résultats sont représentés Fig. 3.41. Dans la première partie du trajet, le vent est dans la direction d'avance du drone. Une erreur statique



(a)



(b)

FIG. 3.38 – (a) chemins parcourus avec différents biais sur la vitesse (coordonnées exprimées en mètres) et (b) erreur sur Y_c (ΔY) exprimée en mètre en fonction du temps (exprimé en secondes).

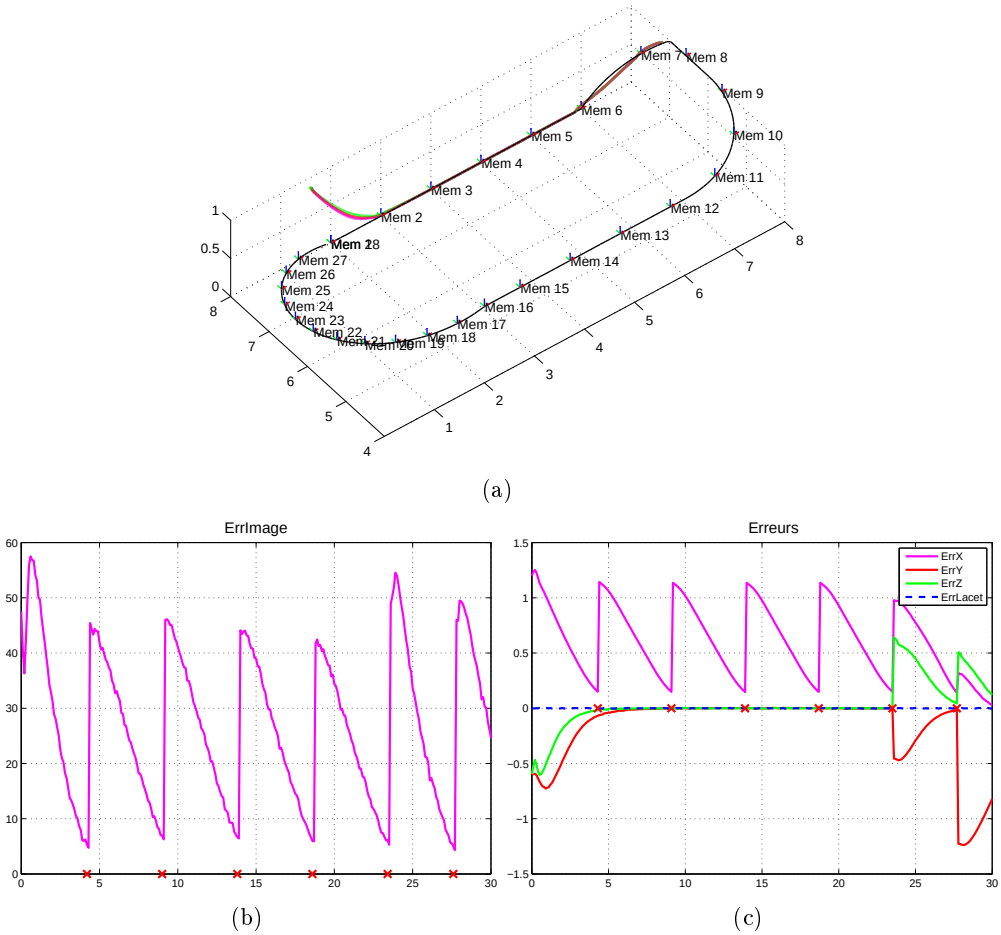
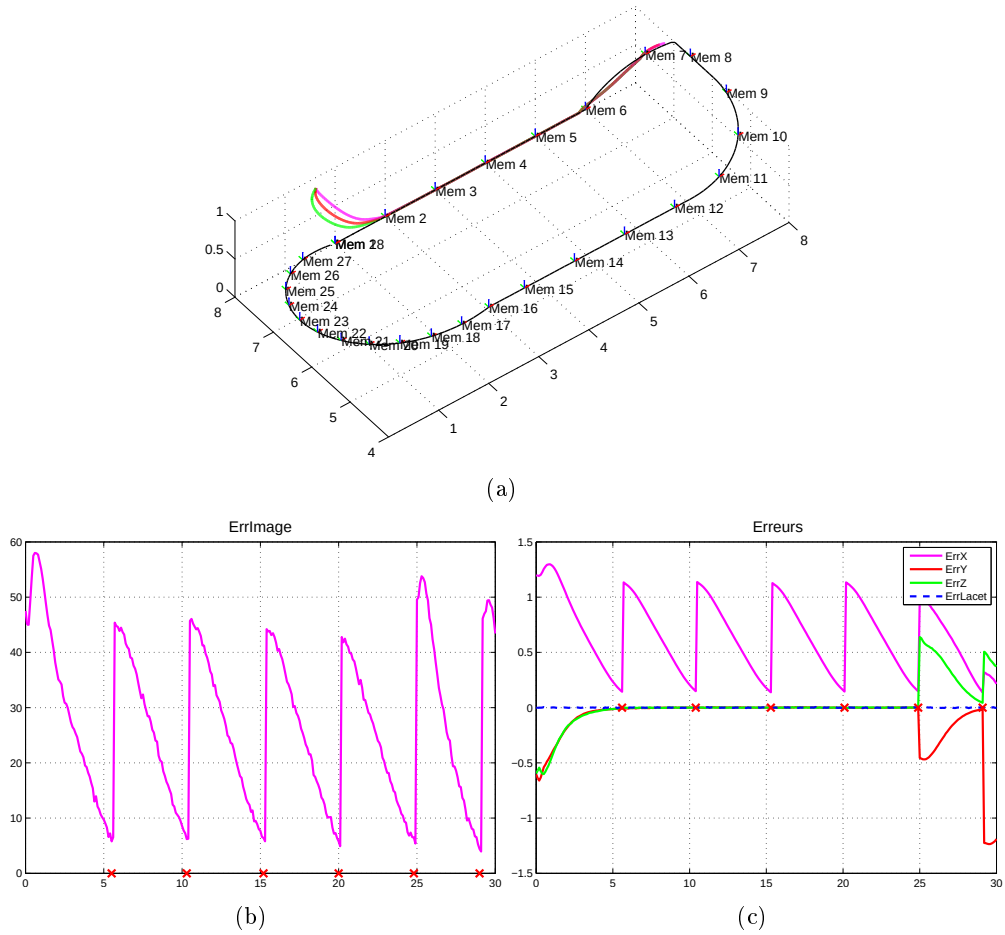


FIG. 3.39 – Perturbation sur le couple Γ_1. (a) chemins parcourus (coordonnées exprimées en mètres) avec plusieurs valeurs de perturbation et (b) erreur dans l'image (en pixels), (c) erreurs de position $\tilde{\mathbf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians en fonction du temps (en secondes) pour une perturbation valant 0.2 rad/s^2.

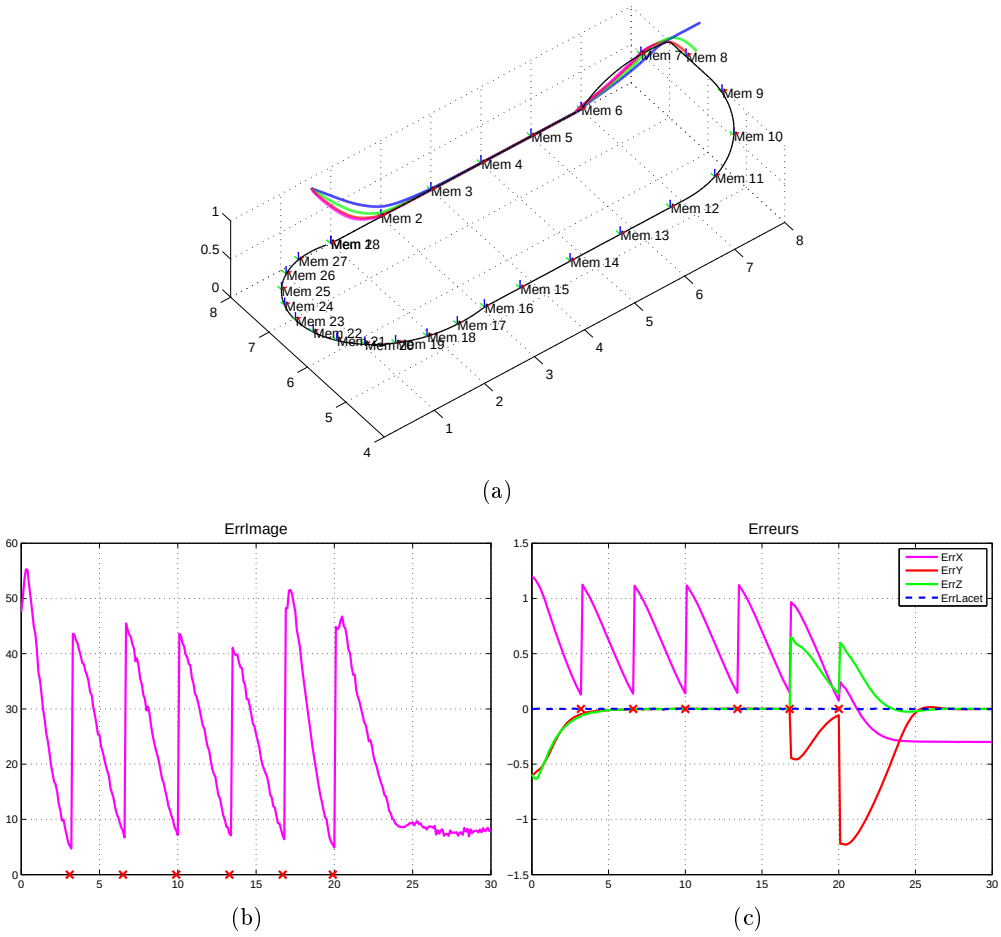


(a)

(b)

(c)

FIG. 3.40 – Perturbation sur le couple Γ_2. (a) chemins parcourus (coordonnées exprimées en mètres) avec plusieurs valeurs de perturbation et (b) erreur dans l'image (en pixels), (c) erreurs de position $\tilde{\mathsf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians en fonction du temps (en secondes) pour une perturbation valant 0.2 rad/s^2.



(a)

(b) (c)

FIG. 3.41 – Perturbation de type vent constant. (a) chemins parcourus (coordonnées exprimées en mètres) avec plusieurs valeurs de perturbation et (b) erreur dans l'image (en pixels), (c) erreurs de position $\tilde{\mathsf{p}}$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians en fonction du temps (en secondes) pour un vent de 1 m/s dans la direction X_w.

sur X_c apparaît. Dans la seconde partie du trajet, la direction du vent et la direction d'avance du drone sont différentes. On observe alors que la tâche de navigation échoue (Fig. 3.41 (a)). Nous avons également réalisé des simulations avec un vent tourbillonnant. Dans ce cas, les résultats obtenus ne sont pas satisfaisants (la tâche de navigation échoue).

Notre approche n'est donc pas robuste vis-à-vis des perturbations de type vent. On peut envisager deux stratégies pour atténuer les effets du vent sur la navigation autonome. La première consiste à mesurer cette perturbation afin de l'introduire dans le schéma de commande. Il s'agit alors de s'interroger sur le dispositif à utiliser, en particulier sur un drone quadrirotor dont la charge utile est très limitée. Une autre stratégie

consiste à établir une loi de commande robuste vis-à-vis de ce type de perturbation.

3.6 Conclusion

Dans ce chapitre, nous avons présenté des lois de commande pour les robots mobiles à roues non-holonomes et pour les drones de type quadrirotor permettant le suivi d'un chemin sensoriel. Dans le premier cas, l'objectif est défini comme le suivi d'un chemin de référence passant par les situations d'acquisition des images clés du chemin sensoriel. Des stratégies basées sur la théorie des systèmes chaînés permettent de réaliser cet objectif sans nécessiter l'ajout de degrés de liberté au capteur. La trajectoire parcourue est alors maîtrisée et le confort pour le matériel et les passagers est pris en compte. Dans le second cas, l'objectif est défini différemment en raison des contraintes de déplacement de ce type d'engin. L'objectif de commande consiste alors à amener le drone aux positions et aux angles de lacet des lieux d'acquisition des images. Cet objectif est réalisé par une stabilisation en position et lacet du drone. L'erreur de position est régulée via un schéma de commande hiérarchique. Les conditions de vol en régime quasi-stationnaire peuvent être remplies et la stabilisation assurée.

De manière générale, les différents résultats de simulation ont montré que les stratégies de suivi de chemin sensoriel sont efficaces et robustes vis-à-vis d'erreurs potentielles de modélisation ou vis-à-vis de bruits de mesure pour les robots mobiles à roues et pour les drones quadrirotors.

De manière plus particulière, concernant les robots à roues, nous avons testé deux critères de changement d'image clé. Le premier (Critère 1) est défini dans l'image et le second (Critère 2) dans l'espace cartésien. Nous avons constaté qu'il était préférable d'employer le Critère 2 pour améliorer globalement la robustesse du système de navigation. Nous avons également observé que dans le cas d'une estimation des translations à un facteur d'échelle près (c'est, par exemple, le cas avec une caméra), les performances de la commande sont garanties si celui-ci est surestimé.

Dans ce même contexte (robots mobiles à roues), nous avons réalisé des simulations pour deux stratégies (définies Section 3.2.1, p. 47). La Stratégie 1 combinée au Critère 1 pour le changement d'image clé a l'avantage d'être d'une mise en œuvre simple dans le sens où elle requiert uniquement l'estimation de l'état partiel $\mathsf{e_p} = [y\,\theta]^\top$. Toutefois, la Stratégie 1 entraîne des discontinuités importantes de $\mathsf{e_p}$ lors des changements d'image clé. La Stratégie 2 permet, par construction, d'éliminer les discontinuités observées sur $\mathsf{e_p}$ au prix d'une complexité de mise en œuvre accrue (en plus de l'état partiel $\mathsf{e_p}$, elle nécessite l'estimation de la courbure et de sa dérivée comme défini dans la Section 3.2.3.2, p. 55).

La stratégie proposée pour le suivi de chemin sensoriel dans le cas d'un drone quadrirotor permet généralement de suivre les chemins appris de façon robuste. On note que la saturation doit être choisie afin d'assurer un bon compromis entre rapidité et sécurité du drone. Dans le cas d'une estimation des positions partielle, il est préférable d'estimer correctement le facteur d'échelle afin de parcourir des chemins sim-

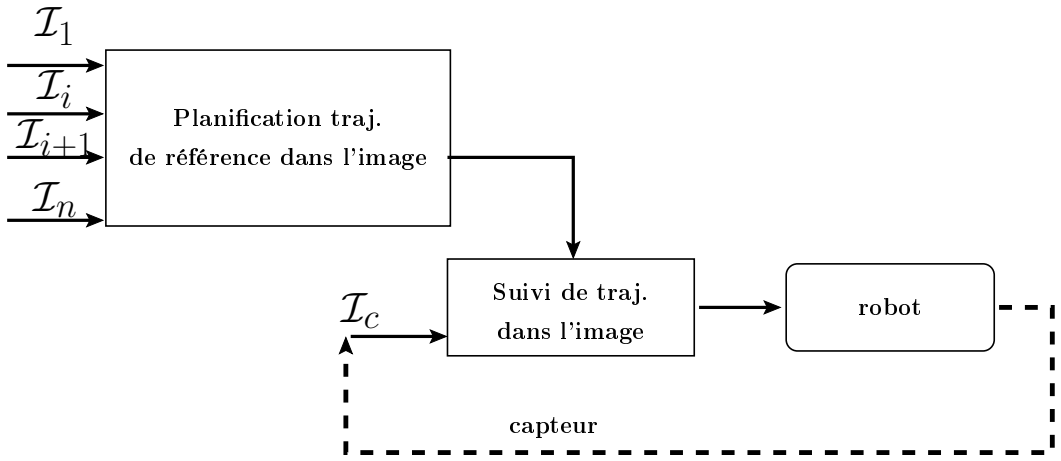


FIG. 3.42 – Schéma bloc d'une stratégie de commande permettant d'éliminer les discontinuités dans les variables d'état et dans la commande en planifiant un chemin de référence dans l'image, admissible par le robot.

ilaires à ceux parcourus lors de l'apprentissage. Cependant, des résultats satisfaisants sont tout de même obtenus dans le cas d'une mauvaise estimation. On observe que les biais sur la vitesse de translation ont une incidence directe sur le résultat du suivi. Il est donc nécessaire de l'estimer correctement. Enfin, le schéma de commande proposé est peu robuste vis-à-vis des perturbations de type vent.

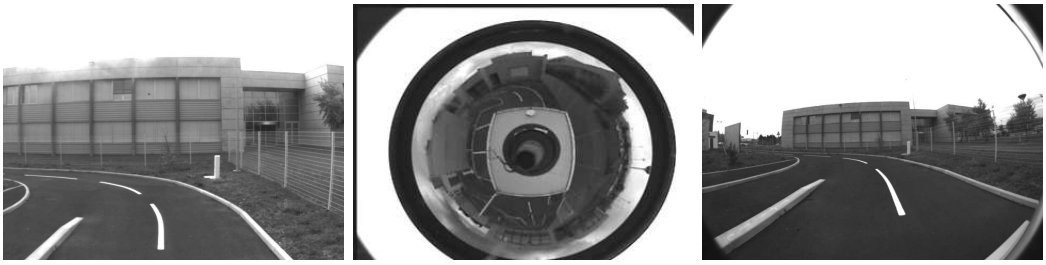
Afin d'améliorer le suivi du chemin sensoriel, une stratégie possible, dans la philosophie des travaux décrits dans [Mezouar 01], consisterait à planifier directement dans l'image un chemin admissible pour le robot (voir Fig. 3.42). De cette façon, les discontinuités du vecteur d'état mais également de la commande pourraient être complètement éliminées. Cette étude reste à mener et constitue une de nos perspectives de recherche.

Chapitre 4

Navigation par mémoire sensorielle : cas d'une caméra grand-angle

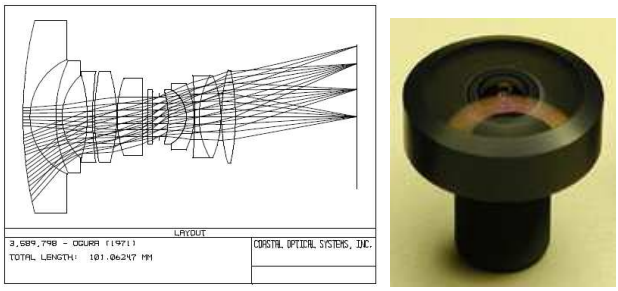
Ces dernières années, la vision panoramique a pris un essor considérable dans de nombreux domaines et notamment en robotique. En effet, les caméras perspectives classiques fournissent un champ de vue restreint de l'environnement dans lequel elles sont plongées, ce qui peut être contraignant dans certaines applications. Trois procédés ont été proposés dans la littérature afin d'accroître le champ de vue des capteurs visuels. Le premier consiste à combiner plusieurs caméras (comme dans [Fermuller 00]) ou à utiliser une caméra en mouvement pour construire une mosaïque. Cette solution a l'avantage de fournir des images panoramiques avec une très grande résolution et l'inconvénient d'être complexe à mettre en oeuvre (étalonnage, synchronisation, ...). Le second procédé consiste à combiner des miroirs à un capteur visuel classique [Baker 98] (caméra catadioptrique). Cette solution a pour avantage la simplicité de mise en œuvre et le temps réduit d'obtention de l'image panoramique. Cependant, d'un point de vue applicatif, les capteurs catadioptriques ont deux inconvénients majeurs : une zone importante au milieu de l'image est inexploitable (voir Fig. 4.1 (b)) et la présence du miroir rend le capteur encombrant et fragile. Enfin, le troisième procédé repose sur l'utilisation d'un objectif à très courte focale ou *lentille fisheye*[1] (voir Fig. 4.1 (c)). Ces objectifs sont généralement construits selon le modèle « rétrofocus » formulé en 1950 par Angénieux qui consiste à coupler une lentille optique divergente avec une lentille convergente (voir Fig. 4.2 (a)). Lorsque l'angle de vue est plus large que la perception de l'oeil humain, un tel capteur est dit grand angle. Cette solution, qui nous intéressera dans la suite, est très attractive du point de vue applicatif car elle permet une observation panoramique et sans zone morte. De plus, les progrès de conception et de fabrication ont permis d'obtenir des lentilles légères, de faibles dimensions et peu onéreuses. La lentille Orifl 190-3 (voir Fig. 4.2 (b)) d'Omnitech Robotics en est un bon exemple : elle mesure 24 mm de diamètre extérieur, pèse environ 25 grammes, coûte 250 dollars et possède un

[1] La dénomination française "objectif hypergone" est peu usitée.



(a) (b) (c)

FIG. 4.1 – Images acquises en environnement urbain avec (a) une caméra perspective, (b) une caméra catadioptrique et (c) une caméra fisheye.



(a) (b)

FIG. 4.2 – (a) Géométrie d'une lentille fisheye (source : Coastal Optic System, 1971). (b) Lentille grand-angle Orifl 190-3 de Omnitech Robotics.

champ de vue supérieur à 180 degrés.

Nous proposons dans ce chapitre de revisiter les trois étapes de notre stratégie de navigation lorsqu'une caméra grand-angle est employée (voir Figure 4.3). L'étape de construction de la mémoire sensorielle MS (Étape 1) a été présentée dans la Section 2.2.1. La première phase de celle-ci consiste à sélectionner les images clés tandis que les deux phases suivantes permettent d'insérer les données capteurs dans la carte sensorielle CS et de mettre à jour les cartes topologiques $CT1$ et $CT0$. Comme ces deux dernières phases sont indépendantes du type de capteur employé, nous nous intéressons ici uniquement à la sélection des images clés décrite dans la Section 4.1 de ce chapitre. L'Étape 2 (localisation initiale) est présentée dans la Section 4.2. Concernant la navigation autonome des robots à roues, nous avons défini deux stratégies de commande (se référer au Chapitre 3). Les lois de commande (3.18) et (3.19), définies pour la Stratégie 1, sont alimentées par le vecteur d'état partiel $\mathsf{e_p} = [y\ \theta]^\top$. Le vecteur $\mathsf{e_p}$ peut être extrait trivialement de la pose $({}^{\mathcal{F}_{i+1}}\mathbf{R}_{\mathcal{F}_c}, {}^{\mathcal{F}_{i+1}}\mathsf{t}_{\mathcal{F}_c})$ du repère courant $\mathcal{F}_c$ associé au véhicule par rapport au repère désiré $\mathcal{F}_{i+1}$ (voir Fig. 4.4 (a)). Pour la Stratégie 2, l'état complet du robot $\mathsf{e} = [s\ y\ \theta]^\top$ est nécessaire afin d'alimenter les lois de commande (3.12) et (3.13). De la même façon que précédemment, celui-ci peut être extrait aisément de la

pose $({}^{\mathcal{F}_{i+1}}\mathbf{R}_{\mathcal{F}_c}, {}^{\mathcal{F}_{i+1}}\mathtt{t}_{\mathcal{F}_c})$ connaissant la trajectoire de référence (voir Fig. 4.4 (b)). Pour les robots aériens, la commande (3.27) est alimentée par l'erreur de position $\tilde{\mathtt{p}}$ et la commande en lacet par $\tilde{\psi}$. La connaissance de la pose $({}^{\mathcal{F}_{i+1}}\mathbf{R}_{\mathcal{F}_c}, {}^{\mathcal{F}_{i+1}}\mathtt{t}_{\mathcal{F}_c})$ permet une nouvelle fois de les estimer aisément (voir Fig. 4.4 (c)). On observe que la transformation $({}^{\mathcal{F}_{i+1}}\mathbf{R}_{\mathcal{F}_c}, {}^{\mathcal{F}_{i+1}}\mathtt{t}_{\mathcal{F}_c})$ peut être obtenue à partir de la transformation $(\mathbf{R}, \mathtt{t})$ liant le repère caméra lors de l'acquisition de l'image $\mathcal{I}_{i+1}$ et le repère caméra courant (image $\mathcal{I}_c$), connaissant les paramètres extrinsèques du capteur (vecteur de translation ${}^{r}\mathtt{t}_c$ et matrice de rotation ${}^{r}\mathbf{R}_c$). Nous verrons dans la Section 4.3.1.2 comment les paramètres du mouvement (la matrice de rotation $\mathbf{R}$ et le vecteur de translation à un facteur d'échelle près $\mathtt{t}_{d^*} = \frac{\mathtt{t}}{d^*}$) peuvent être déterminés pour toute caméra à point central à partir des images $\mathcal{I}_c$ et $\mathcal{I}_{i+1}$ permettant ainsi l'estimation de l'état du robot mobile pour une classe importante de caméras incluant les caméras conventionnelles, les capteurs catadioptriques centraux et, comme nous le détaillerons dans la Section 4.3.2, les caméras fisheye.

4.1 Sélection des images clés

Afin de réaliser la sélection des images clés, nous nous baserons sur les scores d'appariement de primitives entre les images successives de la séquence vidéo acquise lors de la phase d'apprentissage. Comme nous le verrons dans la suite de ce document, l'appariement est d'autant plus une composante majeure de notre système qu'elle sera également utilisée lors de la phase de localisation initiale et lors de la phase de navigation autonome. Dans le paragraphe suivant, nous traitons de la mise en correspondance pour les images grand-angle et présentons l'approche d'appariement de points choisie. Nous décrivons ensuite dans le paragraphe 4.1.2 l'étape de sélection des images clés.

4.1.1 Détection et mise en correspondance dans les images grand-angle

Les stratégies de mise en correspondance peuvent être classées en trois catégories. Dans la première catégorie (approche globale), l'ensemble de l'image est représenté par un descripteur unique. La mise en correspondance est obtenue à partir de la distance ou *similitude* entre les descripteurs. Dans la deuxième catégorie (approche locale), des primitives visuelles sont tout d'abord détectées dans l'image. Chacune de ces primitives est ensuite décrite par un descripteur. Un score de ressemblance est finalement calculé entre deux primitives. La mise en correspondance se base alors sur une sélection des couples en fonction des scores de ressemblance et assez souvent sur des contraintes géométriques. Enfin, dans la dernière catégorie (approche hybride), l'image est partagée en zones globalement décrites. La mise en correspondance se base alors sur les scores de ressemblance entre les descripteurs des mêmes zones.

Pour les images acquises avec une caméra grand-angle, deux stratégies sont envisageables. Dans la première stratégie, les descripteurs sont calculés directement dans l'image panoramique. Il est alors possible d'employer soit des outils classiques de traitement

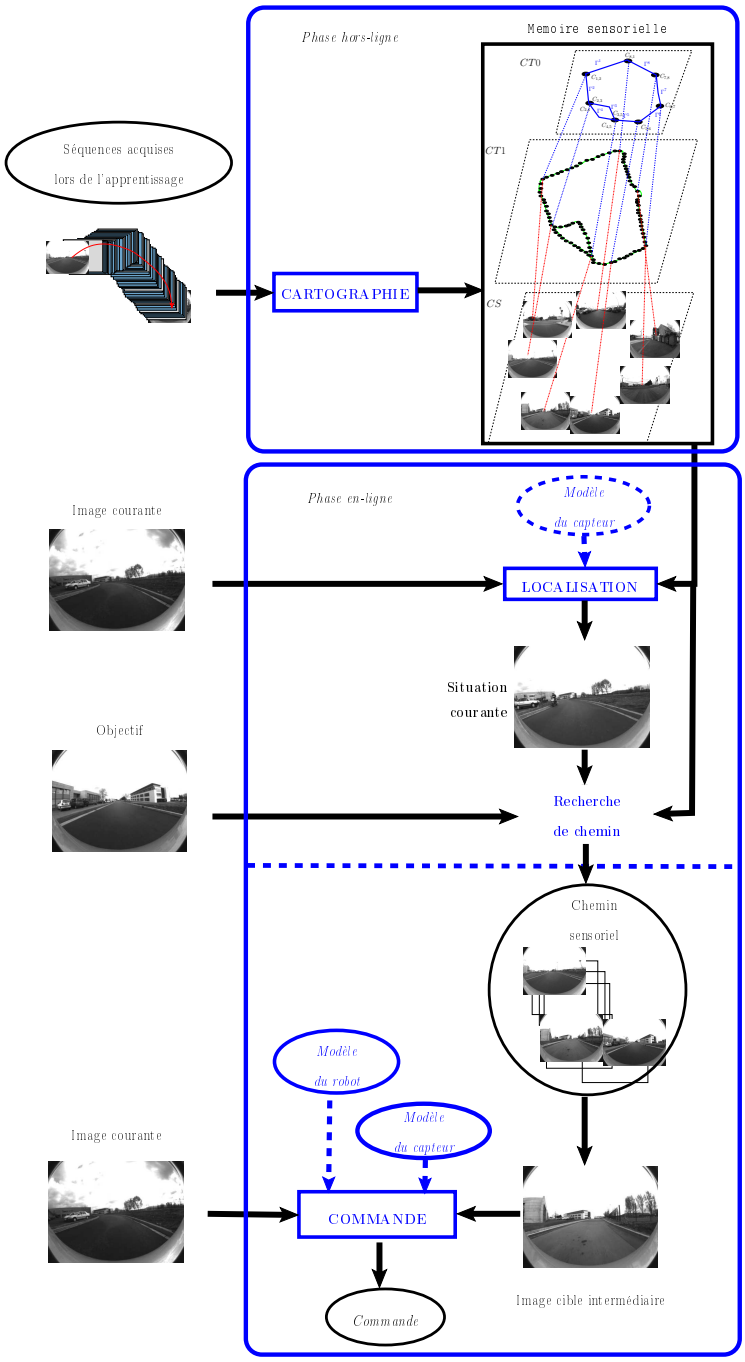
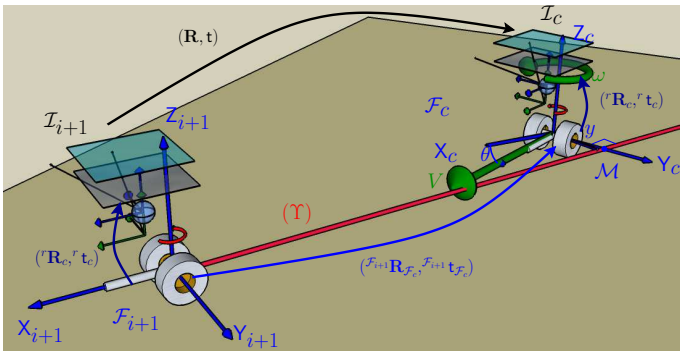
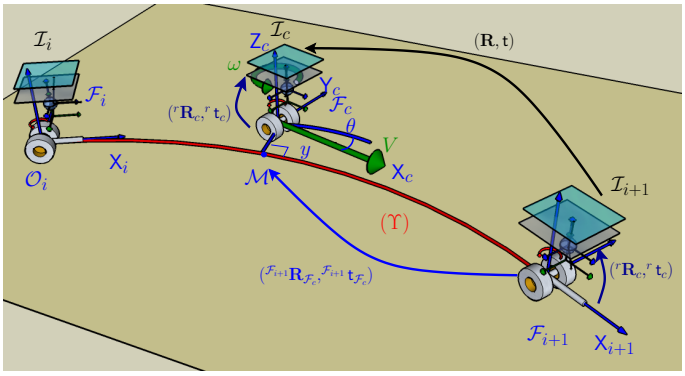


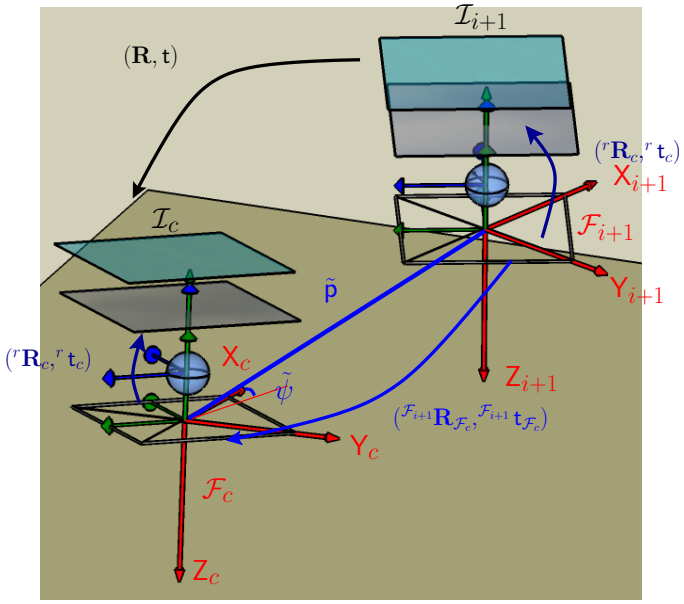
FIG. 4.3 – Notre approche de navigation par mémoire sensorielle appliquée au cas de capteurs visuels.



(a)



(b)



(c)

FIG. 4.4 – Repères pour la commande des robots à roues (a) par la Stratégie 1, (b) par la Stratégie 2 et (c) pour la commande du quadrirotor.

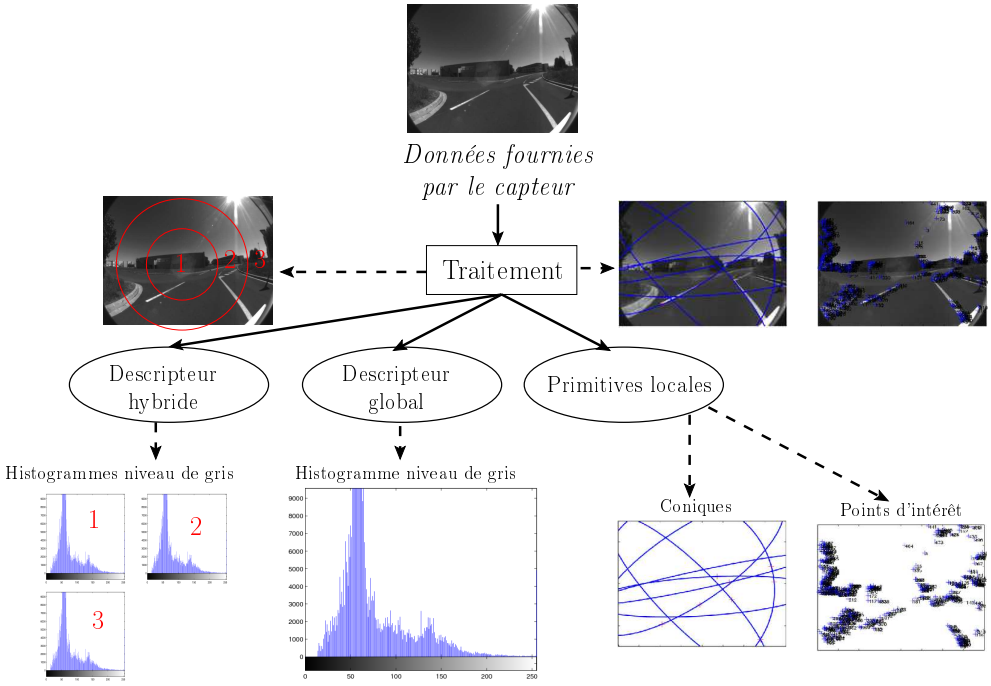


FIG. 4.5 – Représentation des données capteurs acquises avec un capteur visuel grand-angle.

d'images soit des outils adaptés à la géométrie des capteurs visuels omnidirectionnels. Dans la seconde stratégie, employée entre autres dans [Matsumoto 99, Briggs 06], l'image panoramique est projetée sur un cylindre dit *cylindre englobant* et les descripteurs sont calculés sur l'image projetée. L'aspect de cette image est relativement similaire à celle d'une image acquise par une caméra perspective ce qui permet d'utiliser efficacement les outils de traitement d'image classiques. Cependant, cette méthode a un coût calculatoire important en raison de la projection de l'image sur un cylindre et de l'interpolation du contenu de l'image projetée. En outre, elle entraîne une perte d'information (altération due à l'interpolation).

4.1.1.1 Descripteurs

Différents types de descripteurs ont été proposés dans la littérature pour des images acquises avec une caméra grand-angle (voir Fig. 4.5).

Descripteurs globaux Dans [Matsumoto 99], le contenu de l'image est représenté par l'ensemble des niveaux de gris de l'image projetée sur le cylindre englobant. Ce descripteur n'est pas robuste aux mouvements de rotation autour de l'axe optique ni aux changements d'illumination. Afin de rendre ce descripteur invariant à la rotation, l'image peut être orientée dans une direction de référence [Pajdla 99]. Cependant, l'es-

timation de la direction de référence proposée n'est pas robuste aux occultations. Dans [Ulrich 00], le descripteur global de l'image est l'histogramme des niveaux de gris. Ce descripteur est invariant vis-à-vis des rotations autour de l'axe optique mais n'est pas robuste aux changements d'illumination. Une normalisation de l'histogramme permet de surmonter cette difficulté comme il a été proposé, par exemple, dans [Blaer 02] avec des images couleur. Dans [Linåker 04], le descripteur PHLAC (pour *Polar High-order Local AutoCorrelation*) est basé sur 35 fonctions d'autocorrélation adaptées aux coordonnées polaires. Ce descripteur, invariant à la rotation, a un coût calculatoire très faible. Dans [Menegatti 04], les coefficients correspondants aux composantes basse fréquence de la transformée de Fourier forment le descripteur global. Dans [Charron 06], une signature invariante est définie formellement en utilisant les intégrales de Haar.

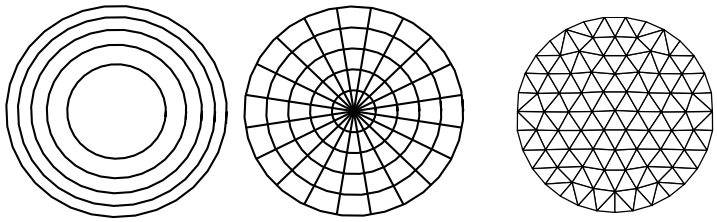
Descripteurs locaux On ne discute ici que des primitives de type points d'intérêt même si d'autres primitives comme les droites (dans [Murillo 07a] par exemple) peuvent être avantageusement utilisées.

Les détecteurs et descripteurs développés pour les caméras conventionnelles sont assez couramment utilisés pour les caméras omnidirectionnelles. Par exemple, le détecteur et descripteur SIFT[2], décrit dans [Lowe 04], a été utilisé entre autres dans [Goedemé 05] pour des images omnidirectionnelles. Cette approche a montré de très bonnes performances avec des images acquises par des caméras classiques et de nombreuses variantes ont été proposées dans la littérature. Parmi celles-ci, un détecteur itératif a été présenté dans [Tamimi 05] pour les images omnidirectionnelles. Le nombre de points d'intérêt obtenus avec ce détecteur est plus faible que celui obtenu avec le SIFT classique. Cela permet de diminuer le coût calculatoire des phases de description et d'appariement. Dans [Murillo 07b], le détecteur et descripteur SURF[3], initialement décrit dans [Bay 08], est utilisé avec des images omnidirectionnelles.

Des descripteurs basés sur des traitements adaptés à la géométrie des caméras omnidirectionnelles ont également été proposés. Dans [Svoboda 01, Ieng 03], les points sont détectés à l'aide d'un détecteur de Harris et Stephens [Harris 88] et la taille et la forme du voisinage utilisé pour la description dépendent de la position du point dans l'image et de la géométrie de la caméra. Dans [Andreasson 05], les points sont détectés dans l'image omnidirectionnelle à partir d'un filtre de Sobel puis décrits à l'aide d'une signature SIFT modifiée. Ce descripteur est calculé sur le voisinage du point d'intérêt qui est orienté en fonction de la position du point dans l'image. Dans [Mauthner 06], une image est construite à partir de la projection de zones d'intérêts de l'image omnidirectionnelle sur un plan tangent à la surface du miroir. Des détecteurs et descripteurs classiques sont ensuite employés sur cette image.

Descripteurs hybrides Dans [Gonzalez-Barbosa 02], l'image est décomposée en anneaux (voir Fig. 4.6(a)) puis chaque anneau est décrit par les histogrammes des dérivées du premier et second ordre des images. Dans [Gaspar 00], l'image projetée sur le cylin-

[2]SIFT : *Scale Invariant Feature Transform*
[3]SURF : *Speeded Up Robust Features*



(a) (b) (c)

FIG. 4.6 – Différents types de décompositions en région pour une image omnidirectionnelle : (a) anneaux, (b) secteurs angulaires, (c) maillage triangulaire.

dre englobant est décomposée en une grille régulière et chaque cellule est ensuite décrite globalement. Dans [Tapus 06], l'image est décomposée en secteurs angulaires. Des primitives sont alors détectées sur chacun d'entre eux. Le descripteur de l'image, appelé *empreinte digitale*, est constitué des primitives ordonnées en fonction du secteur angulaire sur lequel elles se situent.

4.1.1.2 Méthode d'appariement choisie

Il existe, comme nous l'avons vu, de nombreuses méthodes permettant la détection, la description et l'appariement de primitives. Parmi celles-ci, il s'agit de sélectionner la plus pertinente dans notre contexte. Notre principale contrainte est le coût calculatoire lors de la phase de navigation autonome. Il faut également que la méthode choisie permette d'alimenter le processus d'estimation de l'état du système robotique (présenté Section 4.3). Ceci nous a conduit à utiliser le détecteur de Harris et Stephens pour l'extraction des points d'intérêt [Harris 88]. Nous disposons pour celui-ci d'une implémentation optimisée. Ce détecteur est associé à un calcul de corrélation croisée, centrée, normalisée afin de réaliser des appariements de manière très rapide. Plus précisément, nous définissons le support d'un point comme son voisinage centré de taille $(2N+1)\times(2N+1)$. Les supports des points sont ensuite comparés par une mesure de corrélation. Cette corrélation peut être définie comme la somme de la distance pixel à pixel entre les supports. Cependant, cette mesure n'est pas robuste vis-à-vis des changements d'illumination. Pour améliorer ce point, il est préférable d'employer, comme nous l'avons évoqué précédemment, le score de corrélation croisée, centrée, normalisée (ZNCC) qui est donné par :

$$ZNCC(P_i,P_j)=\frac{\sum\limits_{d\in V}\left[\mathcal{I}_i(P_1+d)-\overline{\mathcal{I}_i}(P_1)\right]\left[\mathcal{I}_j(P_2+d)-\overline{\mathcal{I}_j}(P_2)\right]}{\sqrt{\sum\limits_{d\in V}\left[\mathcal{I}_i(P_1+d)-\overline{\mathcal{I}_i}(P_1)\right]^2}\sqrt{\sum\limits_{d\in V}\left[\mathcal{I}_j(P_2+d)-\overline{\mathcal{I}_j}(P_2)\right]^2}} \quad (4.1)$$

avec $\overline{\mathcal{I}_i}(P_i)=\frac{1}{|V|}\sum\limits_{d\in V}\mathcal{I}_i(P_i+d)$, $V=\{-N,\ldots,N\}\times\{-N,\ldots,N\}$ et où $\mathcal{I}_i(P)$ représente le niveau de gris du pixel P dans l'image $\mathcal{I}_i$.

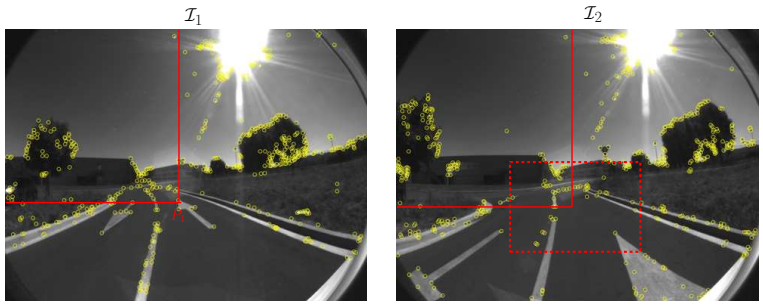


FIG. 4.7 – Fenêtre de recherche pour l'appariement de P_1 dans l'image $\mathcal{I}_2$. Le centre d'un cercle marque la position d'un point d'intérêt.

On définit alors les appariements $(P_1, P_2, ZNCC(P_1, P_2))$ avec P_i un point d'intérêt de l'image $\mathcal{I}_i$. Afin de limiter le coût calculatoire de la phase de mise en correspondance, le score de corrélation entre un point P_1 de l'image $\mathcal{I}_1$ et un point P_2 de l'image $\mathcal{I}_2$ n'est calculé que si P_2 appartient à une certaine zone de l'image. Cette zone de recherche est définie comme le rectangle centré sur la position du point P_1 (voir Fig. 4.7). Tous les appariements $(P_1, P_2, ZNCC(P_1, P_2))$ possibles avec P_2 un point d'intérêt détecté dans la zone de recherche sont formés puis sont retenus si le score de corrélation est supérieur à un certain seuil.

On impose ensuite la contrainte d'unicité : un point d'une image ne peut correspondre qu'à un unique point dans l'autre image. Pour cela, l'appariement $(P_1, P_2, ZNCC(P_1, P_2))$, ayant le score de corrélation le plus élevé, est conservé et les autres appariements contenant P_1 ou P_2 sont éliminés. Ce processus est répété tant qu'il reste des appariements possibles.

Notons que cette méthode peut produire de faux appariements. Les algorithmes utilisant ces points doivent donc être robustes à ce type d'erreur.

4.1.2 Sélection des images clés

Différentes méthodes de sélection des images clés ont été proposées dans la littérature. Celle-ci peut être réalisée manuellement comme dans [Ulrich 00] ou automatiquement. Dans ce cas, l'échantillonnage de la séquence peut être régulier : tous les N mètres parcourus [Gaspar 00], toutes les N secondes [Chen 06], toutes les N images [Hong 91, Zheng 92]. Comme il est noté dans [Kortenkamp 94], ces stratégies peuvent amener à sélectionner un grand nombre d'images inutiles. En outre, la première stratégie nécessite l'utilisation d'un capteur additionnel comme l'odométrie. Il est également possible de se baser sur le nombre de primitives suivies depuis la dernière image sélectionnée [Argyros 05] ou appareillées avec les primitives de la dernière image [Booij 07]. L'échantillonnage obtenu par ces méthodes dépend alors fortement de la robustesse des

algorithmes de détection et de description des primitives. Enfin, il est possible de sélectionner une image lorsqu'une transition est détectée entre deux lieux (porte, intersection de couloirs...) comme dans [Kortenkamp 94].

La représentation par mémoire sensorielle est basée sur la sélection d'images clés. Pour limiter les ressources nécessaires au stockage de la mémoire, le nombre d'images doit être le plus faible possible. Pour cela, le déplacement de la caméra entre deux prises de vue doit être le plus important possible tout en garantissant qu'un nombre suffisant de points puissent être mis en correspondance entre deux images successives. Nous utilisons une méthode heuristique qui, en pratique, donne de très bons résultats. Afin de décrire cette méthode, considérons la séquence d'images $S = \{\mathcal{I}[i] \,|\, i \in \{1, 2, \ldots, n_S\}\}$ acquises lors de la phase d'apprentissage et notons $\{k_j \,|\, j = \{1, 2, \ldots, n\}\}$ $(1 \leq k_j < k_{j+1} \leq n_S)$ les indices des images sélectionnées. La première image de la séquence $\mathcal{I}[1]$ est toujours sélectionnée comme la première image clé $(k_1 = 1)$. Une image clé d'indice k_i est ensuite sélectionnée dans la séquence d'apprentissage S de manière à être la plus éloignée possible de l'image d'indice k_{i-1} et à contenir au moins M points d'intérêt communs avec $\mathcal{I}[k_{i-1}]$. Comme nous le verrons dans le Chapitre 5 dédié aux expérimentations, cette méthode permet d'assurer que suffisamment de primitives sont mises en correspondance entre une image courante $\mathcal{I}_c$ et les deux images qui l'encadrent dans la mémoire visuelle pour l'estimation de l'état du système robotique.

4.2 Localisation initiale

Nous nous intéressons maintenant à la phase de localisation initiale du robot dans sa mémoire sensorielle. Lors de cette étape, le contenu de l'image courante est comparé à celui des images de la carte sensorielle CS afin d'extraire l'image la plus similaire. On rappelle que cette étape est réalisée en ligne. Elle doit donc être un bon compromis entre 1) précision, 2) quantité de données à mémoriser et 3) coût calculatoire.

4.2.1 État de l'art

Le problème de localisation dans une mémoire d'images omnidirectionnelles a été traité de diverses façons dans la littérature. Les méthodes proposées emploient des descripteurs globaux [Menegatti 04, Charron 06] ou des primitives locales [Murillo 07b]. Les méthodes globales ont un coût calculatoire relativement faible mais sont peu robustes vis-à-vis des changements de contenu dans l'image. Au contraire, les approches de localisation locales sont plus robustes mais le coût calculatoire induit est plus élevé comparé aux stratégies globales. Afin de combiner les avantages de ces deux types d'approches, une approche hiérarchique comme celle proposée dans [Murillo 07a] est possible : dans un premier temps, les descripteurs globaux permettent de sélectionner des images candidates tandis que dans une seconde étape, les primitives locales permettent de localiser l'image la plus proche parmi ces images. Le coût calculatoire est alors plus faible que celui d'une approche locale tandis que la robustesse vis-à-vis des changements

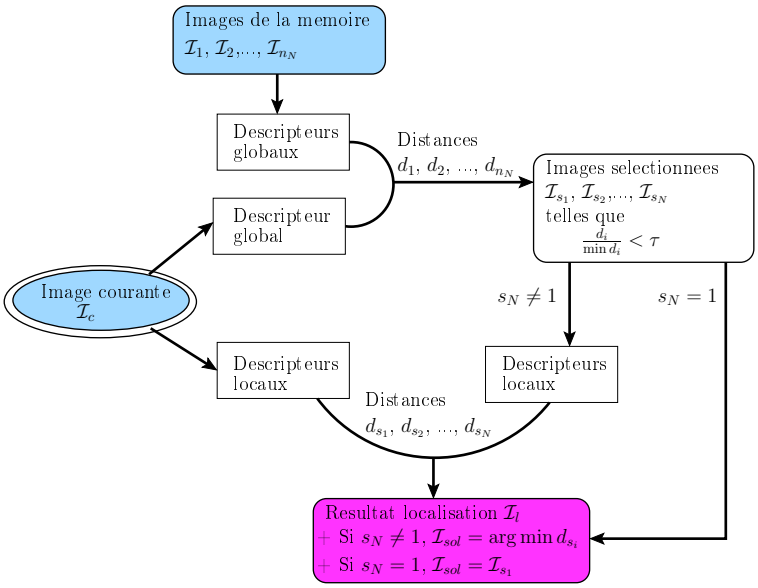


FIG. 4.8 – Approche hiérarchique de localisation.

locaux dans l'image est plus élevée que celle d'une approche globale. La stratégie de localisation décrite dans [Gonzalez-Barbosa 02] est basée sur des descripteurs hybrides. Dans [Charron 06], une approche basée sur les intégrales de Haar a été proposée et comparée à d'autres approches. Cette approche montre de meilleurs résultats pour la localisation de robot en environnement d'intérieur qu'avec les représentations par projection dans l'orientation de référence [Pajdla 99], les signatures basée sur la transformée de Fourier [Menegatti 04] et l'histogramme des niveaux de gris.

4.2.2 Stratégie de localisation proposée

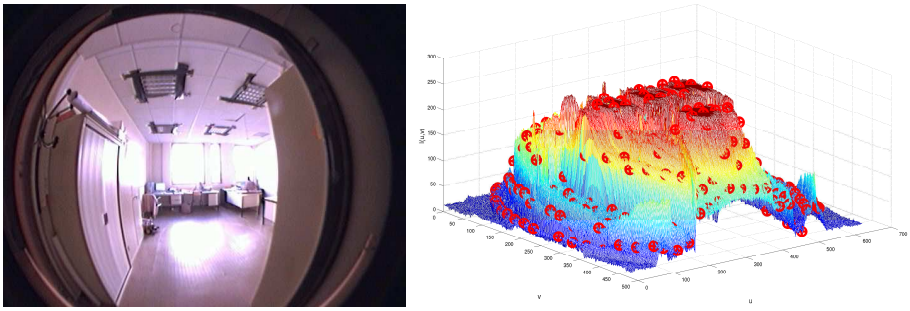
Afin de réduire les temps de calcul d'une méthode locale, une approche hiérarchique semble bien adaptée. En effet, elle permet un traitement relativement plus rapide tout en assurant un certain degré de robustesse vis-à-vis des changements de contenu. Dans cette section, nous détaillons la stratégie globale de localisation adoptée, représentée Figure 4.8.

Une image $\mathcal{I}$ peut être vue comme une surface en trois dimensions où la troisième coordonnée correspond au niveau de gris (voir Fig. 4.9) :

$$\mathcal{I} : \left\{ \begin{array}{ccc} [1,\ldots,N] \times [1,\ldots,M] & \mapsto & [0,255] \\ (u,v) & \rightarrow & \mathcal{I}(u,v) \end{array} \right.$$

où $N \times M$ est la taille de l'image.

L'interpolation consiste alors à approximer localement cette surface $\mathcal{I}(u,v)$ par une surface $f(s,t)$, $s \in [0;1]$, $t \in [0;1]$ passant par des points d'interpolation ou *points de*



(a) (b)

FIG. 4.9 – (a) image omnidirectionnelle et (b) même image vue comme une surface avec les points de contrôle de la surface interpolées (cercles)

contrôle. Plusieurs fonctions d'interpolation peuvent être employées. Afin de limiter le coût calculatoire et de réduire au maximum les erreurs d'interpolation par rapport au contenu initial de l'image, une fonction cubique a été choisie.

Afin que les points de contrôle utilisés soient uniformément positionnés, nous les définissons comme les noeuds d'un maillage triangulaire régulier (voir Fig. 4.6(c)). Un tel maillage peut, par exemple, être obtenu avec le générateur proposé dans [Persson 04]. L'extraction du maillage est alors basée sur l'analogie avec une structure mécanique où les points du maillage correspondent à des noeuds de la structure et les segments à des barres. La position des noeuds est obtenue en amenant la structure mécanique, soumise à des efforts internes dus aux barres et à des efforts externes dus aux frontières de la structure, à l'équilibre.

Afin d'augmenter la robustesse vis-à-vis des changements d'illumination, l'histogramme de l'image est tout d'abord égalisé. Le descripteur global Z est ensuite défini comme l'altitude des points de contrôle de la surface interpolée. La distance d_i entre deux descripteurs Z_c et Z_i correspondant aux images courante $\mathcal{I}_c$ et mémorisée $\mathcal{I}_i$ est choisie comme la distance euclidienne $L1$ entre ces vecteurs, celle-ci ayant donnée les meilleurs résultats sur une série de tests.

Les images sélectionnées sont les images $\mathcal{I}_i$ telles que $\frac{d_i}{d_{min}} \leq \tau$ avec $\tau \in \Re^+$, $\tau \geq 1$ le seuil de sélection et d_{min} la plus petite des distances d_i.

Si, après cette phase, plus d'une image est sélectionnée, alors une localisation locale est réalisée. Celle-ci est basée sur l'approche de mise en correspondance présentée dans le paragraphe 4.1.1.2. La distance entre deux images est alors définie comme l'inverse du nombre nb de primitives appareillées entre l'image courante et la i$^{\text{ème}}$ image candidate ($d_i = d(\mathcal{I}_c, \mathcal{I}_i) = 1/nb$). Le résultat de la localisation est finalement l'image $\mathcal{I}_k$ telle que $d_k = \min_i(d_i)$.

4.2.3 Performances

Dans cette partie, nous présentons tout d'abord les bases de tests ainsi que les différentes approches comparées. L'étude comparative est ensuite menée en terme de quantité de données requises, de précision et de coût calculatoire.

4.2.3.1 Bases de test et méthodes comparées

Trois ensembles d'images omnidirectionnelles sont utilisés : *Almere*, *UAV* et *Walk*. L'ensemble *Almere* a été proposé lors de l'atelier [Data Set 06]. Il contient des images omnidirectionnelles de dimension 1024×768 pixels acquises dans un environnement d'intérieur par une caméra catadioptrique embarquée sur un robot mobile. Quelques images de cet ensemble sont représentées Fig. 4.10(a). Comme il est proposé dans [Murillo 07b], nous n'utilisons qu'une image sur 5 de cet ensemble : la moitié de ces images est utilisée comme référence tandis que l'autre moitié sert d'images tests (978 images). L'ensemble *UAV* contient des images de dimension 384×288 pixels acquises par une caméra fisheye embarquée sur un drone quadrirotor se déplaçant dans un environnement d'intérieur. La caméra est dirigée vers le sol (voir Fig. 4.10(b)), le plafond ou l'avant (voir Fig. 4.10(c)). Les images de références sont sélectionnées toutes les 5 images tandis que les images tests sont sélectionnées toutes les 20 images (188 images). Enfin, l'ensemble *Walk* contient 445 images de dimension 640×480 pixels acquises par une caméra fisheye portée à main d'homme dans des environnements d'intérieur et d'extérieur (voir Fig. 4.10(d) et Fig. 4.10(e)). Contrairement aux autres ensembles, les images tests n'ont pas été acquises en même temps que les images de référence. Les conditions d'illumination, le contenu et les prises de vues correspondant aux images de la mémoire et aux images à localiser sont donc différents.

Le logiciel Matlab a été utilisé afin de comparer les résultats obtenus avec différentes approches. L'approche par interpolation cubique (*Cub*) proposée comme descripteur global (voir le paragraphe précédent) est comparée au PHLAC [Linåker 04] (*PHLAC*) et à l'approche hybride décrite dans [Gonzalez-Barbosa 02] (*Gonz*). Nous comparons également cette approche avec trois approches hybrides. La première consiste à décrire un secteur angulaire de l'image par la moyenne des niveaux de gris (*Sect*). La seconde approche consiste à décrire une région triangulaire de la même façon (*Triang*). Enfin, dans la dernière approche, chaque secteur est représenté par l'histogramme de niveau de gris des pixels de ce secteur (*HistoSect*). Les secteurs angulaires utilisés sont représentés Fig. 4.6 (b) tandis que les secteurs triangulaires sont obtenus à partir des points de contrôle définis pour notre descripteur global (voir Fig. 4.6 (c)). L'approche locale adoptée (*CorrHar*[4]) est comparée aux méthodes SIFT[5] et SURF[6]. Enfin, la méthode

[4]Dans cette approche, nous utilisons 500 points d'intérêt extraits dans l'image et appareillés avec le code C^{++} utilisé dans [Royer 07].

[5]Le détecteur SIFT et le descripteur SIFT de taille 128 sont calculés avec le code de démonstration implémenté en C^{++} disponible sur le site internet de D. Lowe : http ://www.cs.ubc.ca/ lowe/keypoints/http ://www.cs.ubc.ca/ lowe/keypoints/.

[6]Le SURF avec un descripteur de dimension 64 est calculé avec le code C^{++} disponible sur le site

(a) Images de la base *Almere*

(b) Images de la base *UAV* avec la caméra dirigée vers le sol

(c) Images de la base *UAV* avec la caméra dirigée vers l'avant

(d) Images de la base *Walk* en environnement d'intérieur

(e) Images de la base *Walk* en environnement d'extérieur

FIG. 4.10 – Quelques images des différentes bases.

(a) Descripteurs globaux et hybrides

HistoSect : nombre d'éléments	20
Gonz : nombre d'éléments par histogramme	100
Gonz : nombre d'anneaux	5
Seuil sélection approche hiérarchique τ	0.20

(b) Descripteurs locaux

CubHar : nombre de points extraits	500
CubHar : seuil appariement	0.8
SIFT : ratio des distances (plus proche voisin)	0.7
SURF : ratio des distances (plus proche voisin)	0.7

(c) Divers

	Almere	UAV	Walk
Taille image	1024×768	384×288	640×480
Cub et *Triang* : paramètre extraction	0.15	0.25	0.2
Cub et *Triang* : nombre de sommets, nombre de triangle	159, 274	55, 85	88, 143

TAB. 4.1 – Paramètres utilisés lors de nos tests pour les différentes approches de localisation.

hiérarchique finale (*CubCorrHar*) est comparée aux autres approches et également à l'approche hiérarchique utilisant une décomposition par secteurs (*Sect*) combinée avec la méthode (*CorrHar*) (méthode *SectCorrHar*).

Pour toutes les méthodes globales et hybrides, l'histogramme de l'image initiale est tout d'abord égalisé afin d'augmenter la robustesse vis-à-vis des changements d'illumination. Si cela n'est pas précisé, la distance entre deux descripteurs est la norme euclidienne L2. Les paramètres relatifs aux différentes méthodes utilisées sont résumés dans le Tableau 4.1.

4.2.3.2 Taille mémoire requise

Nous comparons tout d'abord les différentes méthodes en terme de taille mémoire requise pour sauvegarder les descripteurs de chaque image. Les résultats sont détaillés dans le Tableau 4.2. Les meilleurs résultats vis-à-vis du critère testé sont en gras tandis que les moins bons sont en italiques.

La taille mémoire requise par les descripteurs locaux est plus importante que celle requise par les descripteurs globaux (voir Tab. 4.2). Les méthodes *Gonz* et *PHLAC* nécessitent très peu de mémoire (0.3 Ko et 0.5 Ko par image) tandis que celle requise par le descripteur cubique reste raisonnable (inférieure ou égale à 1.8 Ko). Ces résultats sont toutefois à nuancer car la taille des descripteurs dépend fortement des paramètres utilisés.

internet des auteurs : http ://www.vision.ee.ethz.ch/ surf/http ://www.vision.ee.ethz.ch/ surf/.

	Almere	*UAV*	*Walk*
Images en niveau de gris	768	108	300
SURF	194	87	150
SIFT	*1100*	*501*	*887*
CorrHar	356	356	356
Gonz	**0.3**	**0.3**	**0.3**
PHLAC	0.5	0.5	0.5
Sect	0.9	0.9	0.9
Triang	1.8	4.0	4.0
Cub	1.5	1.8	1.8

TAB. 4.2 – Taille mémoire requise par image pour différents descripteurs (exprimée en Ko).

Pour l'approche *CorrHar*, le nombre de points détectés est fixe (500) et donc la taille mémoire ne dépend pas de la taille de l'image. Un seuil de détection est, par contre, fixé dans les approches *SIFT* et *SURF* ce qui conduit à un nombre de points détectés dépendant du contenu de l'image. Avec des paramètres de détection et de description ordinaires, le descripteur SIFT nécessite environ cinq fois plus de mémoire que le descripteur SURF.

4.2.3.3 Performances des descripteurs globaux

Les descripteurs globaux sont ici comparés à travers les indicateurs suivants :
- GM est le pourcentage de tests où l'image correcte est trouvée,
- extr. est le nombre moyen d'images sélectionnées (traitées lors de la seconde étape dans une approche hiérarchique),
- GCM est le pourcentage de tests où l'image correcte appartient à l'ensemble des images sélectionnées (le ratio rg=GCM/GCM_{Cub} permet de comparer ce pourcentage à celui obtenu avec le descripteur cubique GCM_{Cub}),
- t représente le coût calculatoire et est comparé au coût obtenu avec le descripteur cubique t_{Cub} à l'aide du ratio rt=t/t_{Cub}.

On observe à travers les résultats synthétisés dans le Tableau 4.3 que l'indicateur GCM de la méthode *PHLAC* est faible (toujours inférieur à 50% des images). Pour les ensembles *Almere* et *Walk*, la méthode *HistoSect* est plus précise en terme d'indicateur GCM (94% et 87% des images). Cependant, le coût calculatoire (indicateur t) est relativement élevé (4.5 secondes). La méthode *Triang* donne de bons résultats pour les ensembles *Almere* et *Walk* mais avec un coût calculatoire plus important que celui de l'approche *Cub*. La méthode par interpolation cubique *Cub* donne de bons résultats pour l'ensemble *UAV* et est un bon compromis entre précision, coût calculatoire et nombre d'images sélectionnées pour les ensembles *Almere* et *Walk*.

	GM(%)	GCM(%)	rg(%)	t	rt	extr.
Almere						
Sect	62.7	62.8	70.5	0.55	4.58	1
HistoSect	90.1	**94.4**	105.9	*4.58*	38.16	*1.26*
Triang	90.5	90.5	101.6	0.38	3.16	1
PHLAC	27.8	*32.7*	36.6	0.47	3.91	*1.27*
Gonz	86.9	88.5	99.3	4.09	34.08	1.07
Cub	89	89.1	100	0.12	1	1
UAV						
Sect	94.1	94.1	96.7	0.1	2.5	1
HistoSect	84	94.1	96.7	*1.45*	36.25	*1.68*
Triang	96.8	**96.8**	99.1	0.11	2.75	1
PHLAC	42	*48.9*	50.2	0.12	3	1.2
Gonz	83.5	86.1	88.5	0.53	13.25	1.11
Cub	97.3	**97.3**	100	**0.04**	1	1
Walk						
Sect	80.8	81.1	96.7	0.28	2	1
HistoSect	69.2	**86.9**	103.7	*4.49*	32.07	*2.82*
Triang	83.8	83.8	100	0.28	2	1
PHLAC	22.6	*28*	33.5	0.25	1.78	1.5
Gonz	54.6	60.8	77.5	1.56	11	1.3
Cub	83.1	83.8	100	**0.14**	1	1

TAB. 4.3 – Performances des descripteurs globaux.

4.2.3.4 Performances des descripteurs locaux

Les performances relatives des approches *SURF*, *SIFT* et *CorrHar* sont ici analysées. Les deux principaux indicateurs utilisés pour cela sont le pourcentage de résultats corrects (GM) mesurant la précision de la méthode et le coût calculatoire t. Le ratio rt=t/$t_{CorrHar}$ permet de comparer ce coût à celui obtenu avec la méthode *CorrHar*. Les performances sont résumées dans le Tableau 4.4.

La méthode *SIFT* donne les meilleurs résultats en terme de précision (indicateur GM) pour les ensembles *Almere* (93.6% de bons résultats) et *Walk* (supérieur à 92% des images) mais le coût calculatoire est très élevé (31 secondes pour la base *Almere* et 152 secondes pour la base *Walk*). Avec les méthodes *SURF* et *CorrHar*, ce coût est environ 7 fois plus faible. L'approche *CorrHar* donne les meilleurs résultats pour l'ensemble *UAV* (plus de 96% des images sont correctement trouvées).

Au vu de ces résultats, l'approche *CorrHar* semble être le meilleur compromis entre précision et coût calculatoire pour les différentes bases testées.

4.2.3.5 Performances de l'approche hiérarchique

Nous comparons ici l'approche hiérarchique proposée (basée sur une interpolation cubique dans un premier temps puis sur la méthode *CorrHar* dans un second temps) aux autres approches. Les résultats sont présentés dans le Tableau 4.5 où les définitions

	GM(%)	t	rt
Almere			
SURF	93.4	**4**	0.43
SIFT	**93.6**	*31.5*	3.46
CorrHar	*91.5*	9.1	1
UAV			
SURF	91.4	1	0.83
SIFT	*90.4*	7.1	5.91
CorrHar	**96.8**	**1.2**	1
Walk			
SURF	*88.7*	**8.5**	0.89
SIFT	**92.5**	*152.1*	16.01
CorrHar	91.2	9.5	1

TAB. 4.4 – Performances des descripteurs locaux.

de GM et t sont identiques à celles données dans la Section 4.2.3.3. La comparaison avec l'approche hiérarchique proposée est effectué à l'aide des ratios : rg=GM/GM$_{CubCorrHar}$ et rt=t/t$_{CubCorrHar}$.

L'approche *SIFT* donne les meilleurs résultats en terme de précision pour les ensembles *Almere* et *Walk* mais a un coût calculatoire très élevé. L'approche *SURF* est un bon compromis entre précision et coût calculatoire pour l'ensemble *Almere* mais est moins précise pour les deux autres ensembles. Si on considère les trois bases d'images, l'approche proposée *CubCorrHar* semble être le meilleur compromis entre coût calculatoire et précision.

Cette section a permis d'introduire la méthode de localisation initiale que nous exploiterons dans notre stratégie de navigation et d'en apprécier les performances. Nous avons notamment constaté qu'elle constitue un très bon compromis entre coût calculatoire, précision et taille mémoire requise. Quelques exemples de résultats de localisation initiale avec les différentes méthodes décrites sont présentés dans l'Annexe A.

La section suivante traite de la tâche de navigation autonome qui peut débuter lorsque le robot est localisé.

4.3 Estimation d'état avec une caméra grand angle

Nous avons vu en introduction que les variables d'état alimentant les lois de commande (3.12), (3.13) et (3.27) peuvent être obtenues à partir du déplacement $(\mathbf{R}, \mathtt{t})$ entre deux prises de vue. Cette section est dédiée à cette problématique. Nous nous focalisons en particulier sur des méthodes de reconstruction partielle suffisamment génériques pour être appliquées à différents types de caméras (conventionelle, catadioptrique et fisheye).

Bien souvent, la géométrie des systèmes de vision omnidirectionnelle ne peut pas être

	GM	$rg(\%)$	t	rt
		Almere		
Sect	62.7	68.5	0.55	0.11
SURF	93.4	102.1	4.06	0.87
SIFT	**93.6**	102.4	*31.5*	6.75
HistoSect	90.1	98.5	4.58	0.98
Cub	89	97.3	**0.12**	0.02
Triang	90.5	99	0.38	0.08
PHLAC	*27.8*	30.4	0.47	0.1
CorrHar	91.5	100.1	9.13	1.95
Gonz	86.9	95	4.09	0.87
SectCorrHar	85.1	93.1	2.07	0.44
CubCorrHar	91.4	100	4.66	1
		UAV		
Sect	94.1	97.2	0.1	0.19
SURF	91.4	94.4	1.04	2.07
SIFT	90.4	93.3	*7.1*	14.14
HistoSect	84	86.7	1.45	2.88
Cub	**97.3**	100.5	**0.04**	0.07
Triang	96.8	100	0.11	0.21
PHLAC	*42*	43.3	0.12	0.23
CorrHar	96.8	100	1.22	2.42
Gonz	83.5	86.2	0.53	1.05
SectCorrHar	95.2	98.3	0.12	0.23
CubCorrHar	96.8	100	0.5	1
		Walk		
Sect	80.8	88.4	0.28	0.05
SURF	88.7	97	8.59	1.69
SIFT	**92.5**	101.2	*152.1*	30.07
HistoSect	69.2	75.7	4.49	0.88
Cub	83.1	90.9	**0.14**	0.02
Triang	83.8	91.6	0.28	0.05
PHLAC	*22.6*	24.7	0.25	0.04
CorrHar	91.2	99.7	9.53	1.88
Gonz	61.1	66.8	1.56	0.3
SectCorrHar	84.9	92.8	0.54	0.1
CubCorrHar	91.4	100	5.05	1

TAB. 4.5 – Performances des approches étudiées.

décrite par un modèle de projection simple [Benosman 00]. Toutefois, lorsque le centre de projection du capteur (point central) est supposé unique (c'est à dire que chaque pixel sur le plan image mesure la luminance du rayon passant par le point central dans une direction particulière et connue) alors un modèle de projection relativement simple est disponible [Geyer 00]. Ces capteurs sont d'un intérêt particulier car, après étalonnage, leurs propriétés géométriques sont similaires à celles des caméras modélisées par une projection perspective. Cela implique que des algorithmes conçus pour les caméras classiques pourront assez aisément être transposés au cas plus général des caméras centrales. Dans [Nayar 97, Baker 98, Baker 99], une liste exhaustive des capteurs à points centraux combinant des miroirs de révolution avec des imageurs conventionnels (caméra catadioptrique) est déduite des propriétés géométriques des systèmes de vision centraux. On y note en particulier qu'une caméra catadioptrique à point central unique peut être obtenue en combinant une caméra classique avec un miroir plan, sphérique,

conique, hyperbolique ou elliptique ou bien en combinant une caméra orthographique avec un miroir parabolique [Baker 98]. En pratique, seules les combinaisons {miroir hyperbolique - caméra perspective} et {miroir parabolique - caméra orthographique} permettent d'obtenir une image omnidirectionnelle [Pajdla 01].

Comme nous l'avons évoqué en introduction de ce chapitre, les caméras fisheye permettent également d'obtenir un champ de vue important. Il est, cependant, communément admis que celles-ci ne rentrent pas dans la catégorie des caméras à point central unique si les différentes distortions observées dans les images sont modélisées. Ces distortions peuvent être classées en trois catégories [Weng 92] : les distortions tangentielles, de décentrage et radiales. Si les distortions tangentielles et les distortions de décentrage sont négligées devant les distortions radiales alors le centre de distortion peut être supposé confondu avec le centre de l'image et le capteur supposé central. Nous verrons que cette hypothèse est tout à fait réaliste dans notre contexte applicatif. Cela nous permettra, de manière remarquable, d'utiliser un modèle de projection unique pour les caméras catadioptriques, conventionelles mais également fisheye. Ainsi, notre stratégie de navigation pourra être utilisée sans modification avec ces trois types de capteurs. Pour cela, nous présentons le modèle de projection unifié dans la section 4.3.1 et nous vérifions son adéquation aux caméras fisheye dans la section 4.3.2.

La géométrie de deux prises de vue, appelée *géométrie épipolaire*, joue un rôle primordial dans de nombreux problèmes relatifs aux caméras à point central unique. Svoboda et Pajdla sont les premiers à avoir étudié la géométrie épipolaire pour les capteurs catadioptriques centraux à miroirs hyperbolique [Svoboda 98] et parabolique [Pajdla 01]. Une formulation générale de cette géométrie pour tout capteur à projection centrale est présentée dans [Svoboda 02]. En particulier, il a été noté que les capteurs à point central unique étalonnés admettent une géométrie épipolaire similaire à celle des caméras conventionnelles [Svoboda 98]. Ce résultat est important car les algorithmes d'estimation des géométries projective et euclidienne, conçus pour les caméras conventionnelles, peuvent être assez directement utilisés pour des capteurs catadioptriques centraux.

4.3.1 Modèle de projection unifié et reconstruction euclidienne

La première partie de cette section couvre la présentation du modèle de projection utilisé dans la suite du document. La seconde partie présente les méthodes mises en œuvre pour la reconstruction Euclidienne à partir de la projection d'un ensemble de points dans une image panoramique.

4.3.1.1 Modèle de projection unifié

La caméra est un capteur qui réalise une projection d'un ensemble d'entités géométriques d'une scène tridimensionnelle dans un espace à deux dimensions : le plan image. Une caméra perspective est décrite par le modèle projectif dit sténopé ou *pinhole*. Une telle caméra réalise une projection perspective, de centre $\mathcal{C}$, de la scène 3D dans le plan image. Afin de décrire plus précisement le modèle sténopé, associons un repère $\mathcal{F}_c(\mathcal{C}, \mathsf{X}_c, \mathsf{Y}_c, \mathsf{Z}_c)$ à la caméra dans sa position courante tel que :

- $\mathcal{C}$ est le centre optique de la caméra,
- Z_c est confondu avec l'axe optique,
- X_c et Y_c sont respectivement parallèles aux lignes et aux colonnes de l'image.

Le modèle pinhole transforme tout point $\mathcal{X}$ de la scène (de coordonnées $\mathsf{X} = [X\ Y\ Z]^\top$ dans $\mathcal{F}_c$), en l'intersection de la droite joignant $\mathcal{X}$ au centre optique avec le plan image normalisé. Les coordonnées homogènes $\mathsf{x}_p = [x_p\ y_p\ 1]^\top$ de ce point d'intersection sont :

$$\mathsf{x}_p = \left[\frac{X}{Z}\ \frac{Y}{Z}\ 1\right]^\top \tag{4.2}$$

Le point dans le plan image réel est obtenu après une transformation homographique $\mathbf{K}_p$:

$$\mathsf{m}_p = [u_p\ v_p\ 1]^\top = \mathbf{K}_p \mathsf{x}_p$$

où :

- $[u_p\ v_p]^\top$ sont les coordonnées du point image exprimées en pixels,
- $\mathbf{K}_p \in \Re^{3\times 3}$ est une matrice contenant les paramètres intrinsèques de la caméra.

Notons maintenant f la focale de la caméra et θ l'angle d'incidence du rayon entrant (c'est-à-dire l'angle entre le rayon $\mathcal{XC}$ et l'axe optique de la caméra). La distance r_p entre le point de l'espace image de coordonnées x_p et le point central (projection du centre optique dans l'image) est donnée par :

$$r_p = f \tan\theta = f\sqrt{x_p^2 + y_p^2} \tag{4.3}$$

Le modèle de projection (4.2) est largement utilisé pour les caméras sans distortion. Pour les capteurs avec distortion comme les caméras catadioptriques, d'autres modèles sont disponibles dans la littérature. On s'intéresse plus particulièrement au modèle unifié de projection proposé dans [Geyer 00] pour les capteurs catadioptriques centraux. Il permet une description simple de l'ensemble des capteurs catadioptriques centraux. Ce modèle repose sur deux projections successives, une première projection centrale sur une sphère virtuelle (induisant une distortion de l'image) suivie d'une projection perspective sur le plan image [Geyer 03, Barreto 03].

Afin de décrire plus précisement ce modèle de projection, considérons une sphère virtuelle unitaire centrée en $\mathcal{M}$ et associons lui le repère $\mathcal{F}_m(\mathcal{M}, \mathsf{X}_m, \mathsf{Y}_m, \mathsf{Z}_m)$ comme indiqué sur la figure 4.11. On peut maintenant supposer qu'une caméra perspective placée en l'origine du repère caméra $\mathcal{F}_c$ observe le monde 3D à travers cette sphère unitaire. Nous supposons dans la suite que $\mathcal{F}_m$ et $\mathcal{F}_c$ sont liés par une translation de $-\xi$ le long de l'axe de l'axe Z_m de $\mathcal{F}_m$. L'origine $\mathcal{M}$ de $\mathcal{F}_m$ est appelée dans la suite centre principal de projection et a pour coordonnées $[0\ \ 0\ \ \xi]^\top$ dans le repère $\mathcal{F}_c$. Soit $\mathsf{X} = [X\ Y\ Z]^\top$ les coordonnées du point 3D $\mathcal{X}$ exprimées dans le repère $\mathcal{F}_m$. Le point $\mathcal{X}$ est projeté dans le plan image en un point de coordonnées homogènes $\mathsf{m} = [u\ v\ 1]^\top$. La formation du point m peut être décomposée en trois étapes :

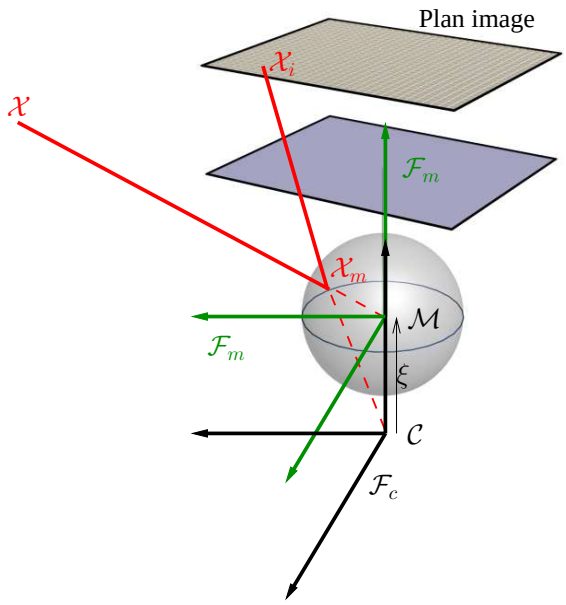


FIG. 4.11 – Modèle de caméra unifié.

Étape 1 : Le point 3D $\mathcal{X}$ est tout d'abord projeté sur la surface de la sphère unitaire en un point $\mathcal{X}_m$ dont les coordonnées X_m exprimées dans le repère $\mathcal{F}_m$ sont :

$$\mathsf{X}_m = \mathsf{X}/\rho \tag{4.4}$$

avec $\rho = \|\mathsf{X}\| = \sqrt{X^2 + Y^2 + Z^2}$.

Étape 2 : Le point sur la sphère $\mathcal{X}_m$ est ensuite projeté perspectivement sur le plan normalisé de coordonnées $Z = 1 - \xi$ en un point de coordonnées homogènes :

$$\mathsf{x} = \begin{bmatrix} \dfrac{X}{Z + \xi\rho} & \dfrac{Y}{Z + \xi\rho} & 1 \end{bmatrix}^\top \tag{4.5}$$

où $\xi \in \Re^+$ dépend du type de caméra.

Étape 3 : Le point dans l'image réelle est finalement obtenu après une transformation homographique $\mathbf{K}$:

$$\mathsf{m} = \mathbf{K}\mathsf{x} \tag{4.6}$$

La matrice $\mathbf{K} \in \Re^{3\times3}$ contient les paramètres intrinsèques de la caméra couplés aux paramètres intrinsèques du système visuel (paramètres du miroir par exemple). Cette matrice $\mathbf{K}$ et le paramètre ξ peuvent être obtenus en utilisant des procédures d'étalonnage comme celles décrites dans [Barreto 03] ou dans [Mei 07].

Notons finalement, que le modèle sténopé peut être obtenu à partir du modèle unifié en posant $\xi = 0$.

Lorsque le capteur est étalonné, les coordonnées du point sur la sphère unitaire peuvent être obtenues à partir des coordonnées du point image en inversant les deux dernières étapes du processus de formation de l'image. Les coordonnées x du point dans le plan image normalisé sont obtenues simplement :

$$\mathsf{x} = [x\ y\ 1]^\top = \mathbf{K}^{-1}\mathsf{m} \tag{4.7}$$

Les coordonnées du point correspondant sur la sphère sont ensuite obtenues en inversant la fonction de projection (4.5) :

$$\mathsf{X}_m = (\eta^{-1} + \xi)\overline{\mathsf{x}} \tag{4.8}$$

$$\overline{\mathsf{x}} = \left[x\ y\ \frac{1}{1+\xi\eta}\right]^\top \tag{4.9}$$

avec : $\begin{cases} \eta = \dfrac{-\gamma - \xi(x^2+y^2)}{\xi^2(x^2+y^2)-1} \\ \gamma = \sqrt{1+(1-\xi^2)(x^2+y^2)} \end{cases}$.

Notons que l'équation (4.8) est vérifiée pour tout point 3D tel que $Z \neq 0$.

4.3.1.2 Contrainte épipolaire

Comme évoqué précédemment, la contrainte épipolaire est une propriété de la projection centrale jouant un rôle primordial dans la reconstruction euclidienne d'un déplacement d'une caméra entre deux prises de vue. Nous en rappelons ici les relations mathématiques fondamentales.

Soit le point 3D $\mathcal{X}$ de coordonnées $\mathsf{X} = [X\ Y\ Z\ ]^\top$ dans le repère $\mathcal{F}_m$ lié à l'image courante $\mathcal{I}_c$ et $\mathsf{X}^* = [X^*\ Y^*\ Z^*\ ]^\top$ dans le repère $\mathcal{F}_m^*$ lié à l'image désirée $\mathcal{I}_{i+1}$ (voir Fig. 4.12). Ces deux repères sont liés par une rotation $\mathbf{R}$ et une translation t. Le plan contenant les centres optiques $\mathcal{C}$ et $\mathcal{C}^*$ ainsi que le point 3D $\mathcal{X}$ est appelé plan épipolaire. On remarque que les points $\mathcal{X}_m$ et $\mathcal{X}_m^*$ (de coordonnées respectives $\mathsf{X_m}$ et $\mathsf{X_m^*}$ dans les repères associés aux sphères) appartiennent également à ce plan (voir Fig. 4.12), ce qui se traduit par la relation :

$$\mathsf{X_m}^\top \mathbf{R}(\mathsf{t} \times \mathsf{X_m^*}) = 0$$

qui peut se réécrire sous la forme :

$$\mathsf{X_m}^\top \mathbf{R}\,[\mathsf{t}]_\times\, \mathsf{X_m^*} = \mathsf{X_m}^\top \mathbf{E} \mathsf{X_m^*} = 0 \tag{4.10}$$

où $[\mathsf{t}]_\times$ représente la matrice antisymétrique relative au vecteur t et $\mathbf{E} = \mathbf{R}\,[\mathsf{t}]_\times$ représente la matrice essentielle [Svoboda 02].

Supposons maintenant que la caméra observe un plan (π) de la scène défini par le vecteur $\pi^{*T} = \left[\mathsf{n}^{*T} - d^*\right]$ où n^* est le vecteur unitaire normal au plan et d^* la distance entre (π) et l'origine de $\mathcal{F}_m^*$. Les coordonnées du point sur la sphère $\mathcal{X}_m$ et les coordonnées du point $\mathcal{X}$ exprimées dans le repère $\mathcal{F}_m^*$ sont liés par la relation :

$$\rho \begin{bmatrix} \mathsf{X_m} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathsf{t} \\ \mathsf{0}_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} \mathsf{X^*} \\ 1 \end{bmatrix} \tag{4.11}$$

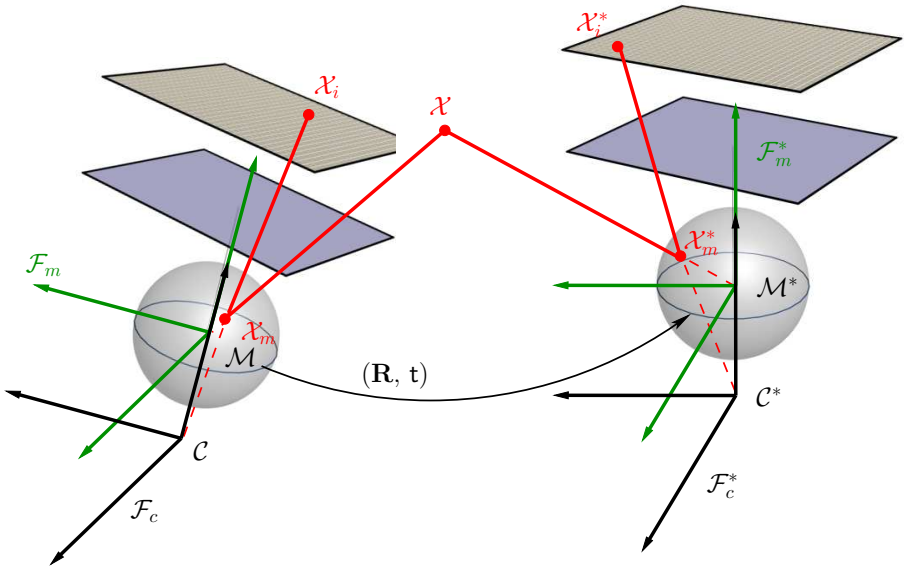


FIG. 4.12 – Géométrie de deux vues avec le modèle unifié sur la sphère.

où $\rho = \sqrt{X^2 + Y^2 + Z^2}$.

En combinant les Équations (4.11) et (4.8), on obtient :

$$\rho(\eta^{-1} + \xi)\overline{\mathsf{x}} = [\mathbf{R} \quad \mathsf{t}]\, \rho^*(\eta^{*-1} + \xi)\overline{\mathsf{x}}^* \tag{4.12}$$

Après quelques manipulations algébriques, on peut écrire :

$$\rho(\eta^{-1} + \xi)\overline{\mathsf{x}} = \rho^*(\eta^{*-1} + \xi)\mathbf{H}_\pi\overline{\mathsf{x}}^* + \alpha\mathsf{t} \tag{4.13}$$

où : $\mathbf{H}_\pi = \mathbf{R} + \frac{\mathsf{t}}{d^*}\mathsf{n}^{*T}$ et $\alpha = -\frac{d(\mathcal{X},\pi)}{d^*}$ avec $d(\mathcal{X}, \pi)$ la distance signée entre le point $\mathcal{X}$ et le plan (π) (indépendante du choix du repère définissant le point et le plan).

$\mathbf{H}_\pi$ est une matrice 3×3 non singulière appelée matrice d'homographie euclidienne relative au plan de référence (π). Cette matrice est fonction des déplacements de la caméra et des coordonnées du plan par rapport au repère $\mathcal{F}_m^*$. Elle a la même forme que dans le cas perspective (elle se décompose en une matrice de rotation et une matrice de rang 1). En outre, si le point $\mathcal{X}$ appartient au plan (π) (*i.e* $\alpha = 0$), on déduit de l'équation (4.13) la relation :

$$\overline{\mathsf{x}} \propto \mathbf{H}_\pi\overline{\mathsf{x}}^* \tag{4.14}$$

L'équation (4.14) peut se ramener en une équation linéaire :

$$\overline{\mathsf{x}} \times \mathbf{H}_\pi\overline{\mathsf{x}}^* = 0 \tag{4.15}$$

où $\times$ représente le produit vectoriel. Comme dans le cas classique, cette matrice peut être estimée à un facteur d'échelle près en utilisant quatre couples de coordonnées $(\mathsf{x}_k, \mathsf{x}_k^*)$, $k = 1..4$ correspondant à la projection dans l'image de points $\mathcal{X}_k$ appartenant

au plan (π). Si seulement 3 points de (π) sont disponibles, cinq couples de points supplémentaires sont nécessaires pour estimer la matrice d'homographie en utilisant, par exemple, l'algorithme linéaire proposé dans [Malis 00].

De notre coté, pour les simulations présentées dans le Chapitre 3, nous avons exploité l'équation (4.15) pour estimer la matrice d'homographie et l'algorithme présenté dans [Faugeras 88] afin d'estimer le déplacement ($\mathbf{R}$, t_{d^*}). Pour les expérimentations réelles qui seront présentées dans le Chapitre 5, nous avons utilisé l'algorithme des cinq points proposé par D. Nistér dans [Nistér 04], couplé à l'algorithme de RanSaC (*Random Sample Consensus*) [Fischler 81] afin de calculer la matrice essentielle de manière robuste. Les paramètres du mouvement ($\mathbf{R}$, t_{d^*}) sont ensuite extraits de $\mathbf{E}$ à partir de sa décomposition SVD (décomposition en valeurs singulières) comme il est décrit dans [Hartley 00].

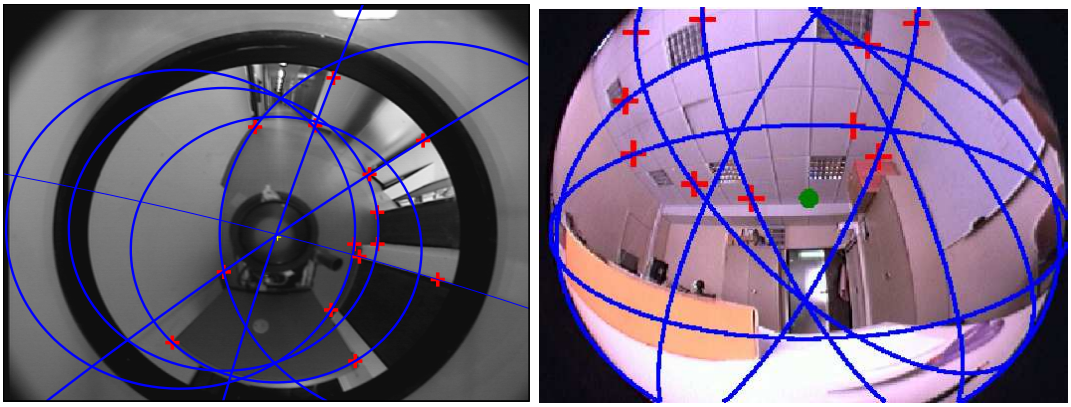
4.3.2 Modèle unifié et caméra fisheye

Le modèle de projection, décrit Section 4.3.1.1, a été développé pour les capteurs catadioptriques centraux. Cependant, comme nous l'avons déjà évoqué, la présence du miroir rend le capteur encombrant, fragile et une partie de l'image inexploitable. L'utilisation d'une caméra fisheye permet de surmonter ces difficultés. Les principaux modèles de projection proposés pour ce type de capteurs peuvent être classés en deux familles : les modèles basés sur une projection perspective et ceux basés sur l'angle d'incidence du rayon entrant. On peut ajouter à ces deux familles principales une troisième famille basée sur le modèle unifié présenté Section 4.3.1.1. Cette famille inclut les propositions décrites dans [Ying 04, Barreto 06, Mei 07]. Celle-ci s'est développée à partir du constat que les images obtenues avec des caméras fisheye et catadioptrique ont de fortes similitudes [Ying 04]. Par exemple, une droite de l'espace se projette en une conique dans le plan image d'un capteur catadioptrique à point central [Barreto 02], ainsi que dans le plan image de certaines caméras fisheye [Smith 99] (voir Fig. 4.13). Dans la suite, nous allons voir que le modèle unifié peut, en fait, être interprété comme un modèle appartenant à la première ou à la seconde des familles mentionnées ci-dessus. Ce résultat est intéressant car il implique que le modèle de projection unifié pourra être utilisé pour les caméras fisheye et, par conséquent, que notre stratégie de navigation pourra être utilisée avec ce type de capteur sans modification.

4.3.2.1 Modélisation des caméras fisheye

Nous nous intéressons ici uniquement aux modèles de distortions radiales. Le centre de distortion étant supposé confondu avec le point origine du plan image, ces distortions influent uniquement sur la distance r_f entre le point de l'espace image de coordonnées projectives x et le point principal (intersection de l'axe optique avec le plan image).

Les modèles de distortions radiales proposés dans la littérature sont nombreux mais ils respectent tous deux contraintes fondamentales :



(a) (b)

FIG. 4.13 – Estimation des paramètres de la projection de droites dans l'image (a) d'une caméra catadioptrique centrale, (b) d'une caméra fisheye, en utilisant le modèle unifié et deux points cliqués à la souris sur chaque droite.

Contrainte 1 : le rayon arrivant le long de l'axe optique n'est pas déformé.

Contrainte 2 : la distance r_f croit de façon monotone lorsque l'angle d'incidence augmente.

Pour les caméras avec distortions, la distance entre le centre de l'image et le point image est différente de celle obtenue dans le cas sténopé :

$$r_f = r_f(\theta) \neq r_p(\theta) = f \tan\theta$$

La première famille de modèles repose sur le modèle sténopé. Le rayon r_f est obtenu en ajoutant des distortions radiales au rayon r_p (projection de type T_1) :

$$r_p \xrightarrow[T_1]{} r_f$$

D'après la définition de r_p, ces modèles ne sont pas définis lorsque l'angle d'incidence est égal à $\frac{\pi}{2}[\pi]$.

La seconde famille de modèles repose sur une transformation entre l'angle d'incidence et la distance r_f (projection de type T_2) :

$$\theta \xrightarrow[T_2]{} r_f$$

Quelques modèles de projections de type T_1 sont présentés dans le Tableau 4.6. Dans [Pajdla 97], le modèle de distortion polynomial $r_f^1(r_p)$ est utilisé. Ce modèle est classiquement employé pour modéliser les distortions d'une caméra perspective réelle [Zhang 98]. Peu de paramètres sont nécessaires lorsque le champ de vue est restreint mais la modélisation des distortions des caméras grand-angle nécessitent de nombreux

$$
\begin{array}{c}
r_f^1(r_p) = r_p L(r_p, n) \\
r_f^2(r_p) = \dfrac{r_p}{L(r_p, n)} \\
r_f^3(r_p) = \dfrac{r_p}{1 + k_1 r_p^2} \\
r_f^4(r_p) = r_p \dfrac{L_1(r_p, n_1)}{L_2(r_p, n_2)} \\
r_f^5(r_p) = s \log(1 + \lambda r_p),\ (s,\ \lambda) \in \Re^{+2} \\
r_f^6(r_p) = \dfrac{1}{\omega} \arctan\left(2 r_p \tan\dfrac{\omega}{2}\right),\ \omega \in [0,\ 2\pi]
\end{array}
$$

TAB. 4.6 – Principaux modèles de caméra fisheye basés sur une projection perspective. Le polynôme $L(r_p, n)$ est du type : $1 + k_1 r_p^2 + k_2 r_p^4 + \cdots + k_n r_p^{2n},\ (k_1,\ k_2, \ldots, k_n) \in \Re^n$.

paramètres. A. Fitzgibbon a proposé le modèle de division $r_f^2(r_p)$ [Fitzgibbon 01] qui peut être utilisé avec un paramètre unique ($r_f^3(r_p)$) même pour un champ de vue important. La projection peut également être représentée par le modèle rationnel $r_f^4(r_p)$ ($n_1 + n_2$ paramètres) [Li 05], le modèle logarithmique $r_f^5(r_p)$ [Basu 95] (contenant un paramètre d'échelle s et un paramètre λ permettant de corriger la distortion) ou par le modèle de champ de vue (FOV pour *Field-Of-View*) $r_f^6(r_p)$ [Devernay 01].

Les principales fonctions de projection basées sur l'angle d'incidence (type T_2) sont présentées dans le Tableau 4.7. La fonction la plus classique est la projection f-theta (également appelée projection linéairement divisé ou projection équidistante) $r_f^1(\theta)$ proposée dans [Kingslake 89]. Ce modèle de projection est, cependant, limitée aux caméras avec de faibles distortions. La projection stéréographique $r_f^2(\theta)$ proposée dans [Fleck 94, Stevenson 95] préserve la circularité. Le modèle orthographique $r_f^3(\theta)$ a été proposé dans [Ray 94] et le modèle à angle équisolide (ou à surface fidèle) $r_f^4(\theta)$ dans [Smith 92]. La projection polynomiale $r_f^5(\theta)$ [Xiong 97, Kannala 06, Scaramuzza 06] permet d'améliorer la précision par rapport au modèle $r_f^1(\theta)$. Des modèles hybrides ont également été étudiés : $r_f^6(\theta)$ avec α un facteur d'échelle et β le paramètre de projection radiale [Kumler 00] et $r_f^7(\theta)$ qui est une combinaison du modèle stéréographique (de paramètres a, b) et de la projection d'angle équisolide (de paramètres c, d) [Bakstein 02].

4.3.2.2 Équivalences avec le modèle unifié

Dans le cas de la projection perspective, la distance r_p (équation 4.3)) peut se réécrire sous la forme : $r_p = \frac{f}{Z}\sqrt{X^2 + Y^2}$ comme $Z > 0$.

$$r_f^1(\theta) = f\theta$$
$$r_f^2(\theta) = 2f\tan\left(\frac{\theta}{2}\right)$$
$$r_f^3(\theta) = f\sin\theta$$
$$r_f^4(\theta) = f\sin\left(\frac{\theta}{2}\right)$$
$$r_f^5(\theta) = f(k_1\theta + k_2\theta^3 + \cdots + k_n\theta^{2(n-1)+1}),\ (k_1,\ k_2, \ldots, k_n) \in \Re^n$$
$$r_f^6(\theta) = \alpha\sin(\beta\theta)$$
$$r_f^7(\theta) = a\tan(\theta/b) + c\sin(\theta/d)$$

TAB. 4.7 – Principaux modèles de caméra fisheye basés sur l'angle d'incidence

On considère maintenant la projection avec le modèle unifié. Les coordonnées du point projeté sont :

$$\begin{cases} x_f = \frac{fX}{Z+\xi\sqrt{X^2+Y^2+Z^2}} \\ y_f = \frac{fY}{Z+\xi\sqrt{X^2+Y^2+Z^2}} \end{cases} \tag{4.16}$$

À partir de ces coordonnées et de l'Éq. (4.3), on obtient la relation suivante :

$$r_f^2 = \frac{r_p^2}{\left(1+\xi\sqrt{\left(\frac{x_p}{f}\right)^2+\left(\frac{y_p}{f}\right)^2+1}\right)^2} \tag{4.17}$$

L'équation (4.17) peut alors se réécrire sous la forme :

$$r_f = r_f(r_p) = \frac{r_p}{1+\xi\sqrt{\frac{r_p^2}{f^2}+1}} \tag{4.18}$$

L'équation (4.18) est une transformation de type T_1 qui relie la distance r_p à la distance r_f. On vérifie facilement que lorsque le rayon d'incidence arrive le long de l'axe optique alors $r_p = 0$ et la distance entre le centre optique et le point projeté est donc nulle ($r_f(0) = 0$). La Contrainte 2 est également respectée.

En utilisant l'équation (4.18) et comme $r_p = f\tan\theta$, on obtient directement la relation :

$$r_f = r_f(\theta) = \frac{f\tan\theta}{1+\xi\sqrt{\tan^2\theta+1}} \tag{4.19}$$

L'équation (4.19) est une transformation de type T_2 reliant la distance r_f et l'angle d'incidence θ. On vérifie facilement que lorsque le rayon d'incidence arrive le long de

l'axe optique ($\theta = 0$), la distance entre le centre optique et le point projeté est nulle ($r_f(0) = 0$) et que la Contrainte 2 est respectée.

On remarque de manière classique que le modèle présenté sous cette forme n'est pas valable pour $\theta = \pi/2$ et :

$$\lim_{\substack{\theta \to \pi/2 \\ \theta < \pi/2}} r(\theta) = \frac{f}{\xi}$$

Nous ne pouvons pas conclure directement à partir des équations (4.18) et (4.19) que le modèle unifié permet de modéliser efficacement les caméras fisheye (tout comme pour l'ensemble des modèles proposés dans la littérature). De manière expérimentale, nous pouvons, cependant, nous faire une idée sur ce point. À titre d'exemple, un résultat d'estimation des paramètres de projection de droites dans l'image d'une caméra fisheye en utilisant le modèle unifié et deux points cliqués à la souris sur chaque droite est représenté Fig. 4.13 (b). On constate visuellement que le résultat est correct. Afin de s'assurer que ce modèle est suffisant pour les applications robotiques envisagées, nous proposons dans la suite des résultats plus quantitatifs d'étalonnage et de reconstruction euclidienne.

4.3.2.3 Étalonnage

Nous proposons, tout d'abord, d'étalonner quatre caméras à l'aide d'une mire plane. Trois modèles de projection sont comparés :

- le modèle polynomial basé sur la projection perspective : $r_f^1(r_p) = r_p(1 + a_1 r_p^2 + \cdots + a_3 r_p^6)$,
- le modèle unifié auquel des paramètres de distortions radiales sont ajoutés[7] : $r_f(r_c) = r_c(1 + a_1 r_c^2 + \cdots + a_3 r_c^6)$ (avec r_c la distance obtenue avec le modèle unifié),
- le modèle unifié.

Les quatre capteurs considérés sont des caméras munies de lentilles :

- Pentax TS212A (C70214) (champ de vue de 94 degrés). Les images sont de taille 640×480 pixels.
- Sigma 8/3,5 EX DG [8] (champ de vue de 180 degrés). Les images sont de taille 1020×1020 pixels.
- Fujinon Fisheye E5 1 :1.4/1.4mm (champ de vue de 185 degrés). Les images sont de taille 1024×768 pixels.
- Orifl 190-3 d'Omnitech Robotics (champ de vue de 190 degrés). Les images sont de taille 640×480 pixels.

[7] Ce modèle a été proposé dans [Mei 07]. Nous négligeons ici les paramètres de distortions tangentielles.

[8] Contrairement aux autres lentilles, celle-ci a été montée sur une caméra analogique et les images brute directement acquises par le capteur sont employées.

	Modèle $r_f^1(r_p)$	Modèle unifié avec distortions	Modèle unifié
Distances focales : $\begin{pmatrix} f_u \\ f_v \end{pmatrix}$	$\begin{pmatrix} 384.2 \\ 383.8 \end{pmatrix}$	$\begin{pmatrix} 384.5 \\ 384.2 \end{pmatrix}$	$\begin{pmatrix} 384.3 \\ 383.9 \end{pmatrix}$
Point principal : $\begin{pmatrix} u_0 \\ v_0 \end{pmatrix}$	$\begin{pmatrix} 343.7 \\ 268.0 \end{pmatrix}$	$\begin{pmatrix} 343.7 \\ 267.9 \end{pmatrix}$	$\begin{pmatrix} 343.5 \\ 268.2 \end{pmatrix}$
Paramètres :	$a = \begin{pmatrix} -0.284 \\ 0.145 \\ -0.050 \end{pmatrix}$	$a = \begin{pmatrix} -0.289 \\ 0.490 \\ -0.884 \end{pmatrix}$ $\xi = 0.581$	$\xi = 1.253$
Err. reproj. : $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 0.197 \\ 0.156 \end{pmatrix}$	$\begin{pmatrix} 0.194 \\ 0.156 \end{pmatrix}$	$\begin{pmatrix} 0.202 \\ 0.143 \end{pmatrix}$
Err. reproj. $Val.$ $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 0.210 \\ 0.164 \end{pmatrix}$	$\begin{pmatrix} 0.205 \\ 0.167 \end{pmatrix}$	$\begin{pmatrix} 0.203 \\ 0.209 \end{pmatrix}$

TAB. 4.8 – Étalonnage d'une caméra munie d'une lentille Pentax TS212A (C70214). Cette caméra a un champ de vue de 94 degrés. La taille de l'image est fixée à 640×480 pixels.

L'étalonnage des caméras est réalisé en utilisant les toolbox Matlab de Caltech[9] pour le modèle perspective et de C. Mei[10] pour les modèles unifiés. La matrice des paramètres intrinsèques de la caméra est :

$$\mathbf{K_p} = \begin{pmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.20}$$

Selon les lentilles, les points d'étalonnage sont extraits sur 5 à 7 images d'une grille. Les mêmes points sont utilisés pour les trois modèles. Pour chaque caméra et chaque modèle de projection, l'erreur moyenne de reprojection sur l'image $[err_u \; err_v]^\top$, exprimée en pixels, est calculée. Nous estimons également l'erreur de reprojection annotée $Val.$ des points d'une image mise de côté lors de l'étalonnage, cela afin de valider les valeurs des paramètres intrinsèques du capteur.

Les résultats sont présentés dans les Tableaux 4.8 pour la TS212A (C70214) de Pentax, 4.9 pour la Sigma 8/3,5 EX DG de Canon, 4.10 pour la Fisheye E5 de Fujinon et 4.11 pour l'objectif Orifl190-3 d'Omnitech Robotics.

Pour les deux premiers capteurs, les résultats sont similaires avec les différents modèles de projection, que ce soit pour les distances focales, la position du point principal ou les erreurs de reprojection. Lorsque le champ de vue devient supérieur à 180 degrés, les erreurs de reprojection données par le modèle $r_f^1(r_p)$ ne sont pas satisfaisantes (plus de 2 pixels sur chaque axe pour la lentille Fujinon et plus de 50 pixels pour la lentille ORIFL). Le modèle unifié est alors plus apte à modéliser les fortes distortions. On observe que celles-ci sont bien représentées par le paramètre ξ du modèle unifié et que les paramètres de distortion additionnels apportent peu de précision supplémentaire ($[0.197 \; 0.156]^\top$ contre $[0.156 \; 0.174]^\top$ pour la lentille Fujinon), une précision similaire

[9] Toolbox de Caltech : http ://www.vision.caltech.edu/bouguetj/calib_doc/http ://www.vision.caltech.edu/bouguetj/calib_doc/

[10] La toolbox de Mei est disponible sur son site internet : http ://www.robots.ox.ac.uk/ cmei/Toolbox.htmlhttp ://www.robots.ox.ac.uk/ cmei/Toolbox.html.

	Modèle $r_f^1(r_p)$	Modèle unifié avec distortions	Modèle unifié
Distances focales : $\begin{pmatrix} f_u \\ f_v \end{pmatrix}$	$\begin{pmatrix} 347.5 \\ 347.1 \end{pmatrix}$	$\begin{pmatrix} 347.7 \\ 347.3 \end{pmatrix}$	$\begin{pmatrix} 351.0 \\ 350.5 \end{pmatrix}$
Point principal : $\begin{pmatrix} u_0 \\ v_0 \end{pmatrix}$	$\begin{pmatrix} 502.5 \\ 525.4 \end{pmatrix}$	$\begin{pmatrix} 502.4 \\ 525.4 \end{pmatrix}$	$\begin{pmatrix} 502.2 \\ 525.4 \end{pmatrix}$
Paramètres :	$a = \begin{pmatrix} -0.329 \\ 0.132 \\ -0.027 \end{pmatrix}$	$a = \begin{pmatrix} 0.704 \\ -0.672 \\ 37.659 \end{pmatrix}$ $\xi = 3.405$	$\xi = 2.556$
Err. reproj. : $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 0.115 \\ 0.105 \end{pmatrix}$	$\begin{pmatrix} 0.110 \\ 0.102 \end{pmatrix}$	$\begin{pmatrix} 0.100 \\ 0.099 \end{pmatrix}$
Err. reproj. $Val.$ $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 0.178 \\ 0.187 \end{pmatrix}$	$\begin{pmatrix} 0.218 \\ 0.205 \end{pmatrix}$	$\begin{pmatrix} 0.153 \\ 0.168 \end{pmatrix}$

TAB. 4.9 – Étalonnage d'une caméra munie d'une lentille Sigma 8/3,5 EX DG. Cette caméra a un champ de vue de 180 degrés. La taille de l'image est fixée à 1020×1020 pixels.

	Modèle $r_f^1(r_p)$	Modèle unifié avec distortions	Modèle unifié
Distances focales : $\begin{pmatrix} f_u \\ f_v \end{pmatrix}$	$\begin{pmatrix} 243.7 \\ 239.5 \end{pmatrix}$	$\begin{pmatrix} 208.6 \\ 208.7 \end{pmatrix}$	$\begin{pmatrix} 208.5 \\ 208.5 \end{pmatrix}$
Point principal : $\begin{pmatrix} u_0 \\ v_0 \end{pmatrix}$	$\begin{pmatrix} 500.0 \\ 372.8 \end{pmatrix}$	$\begin{pmatrix} 521.9 \\ 391.5 \end{pmatrix}$	$\begin{pmatrix} 521.7 \\ 390.9 \end{pmatrix}$
Paramètres :	$a = \begin{pmatrix} -0.201 \\ 0.028 \\ -0.001 \end{pmatrix}$	$a = \begin{pmatrix} 0.066 \\ 0.173 \\ -0.324 \end{pmatrix}$ $\xi = 1.720$	$\xi = 1.586$
Err. reproj. : $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 1.824 \\ 2.265 \end{pmatrix}$	$\begin{pmatrix} 0.142 \\ 0.148 \end{pmatrix}$	$\begin{pmatrix} 0.130 \\ 0.136 \end{pmatrix}$
Err. reproj. $Val.$ $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 2.787 \\ 4.396 \end{pmatrix}$	$\begin{pmatrix} 0.156 \\ 0.174 \end{pmatrix}$	$\begin{pmatrix} 0.174 \\ 0.205 \end{pmatrix}$

TAB. 4.10 – Étalonnage d'une caméra munie d'une lentille Fujinon Fisheye E5 1 :1.4/1.4mm. Cette caméra a un champ de vue de 185 degrés. La taille de l'image est fixée à 1024×768 pixels.

	Modèle $r_f^1(r_p)$	Modèle unifié avec distortions	Modèle unifié
Distances focales : $\begin{pmatrix} f_u \\ f_v \end{pmatrix}$	$\begin{pmatrix} 291.5 \\ 293.5 \end{pmatrix}$	$\begin{pmatrix} 223.3 \\ 222.5 \end{pmatrix}$	$\begin{pmatrix} 222.9 \\ 222.1 \end{pmatrix}$
Point principal : $\begin{pmatrix} u_0 \\ v_0 \end{pmatrix}$	$\begin{pmatrix} 315.6 \\ 255.5 \end{pmatrix}$	$\begin{pmatrix} 305.3 \\ 266.9 \end{pmatrix}$	$\begin{pmatrix} 305.1 \\ 266.9 \end{pmatrix}$
Paramètres :	$a = \begin{pmatrix} -0.427 \\ 0.142 \\ -0.017 \end{pmatrix}$	$a = \begin{pmatrix} 1.385 \\ -12.860 \\ 572.772 \end{pmatrix}$ $\xi = 4.552$	$\xi = 2.854$
Err. reproj. : $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 2.013 \\ 1.933 \end{pmatrix}$	$\begin{pmatrix} 0.158 \\ 0.132 \end{pmatrix}$	$\begin{pmatrix} 0.157 \\ 0.141 \end{pmatrix}$
Err. reproj. $Val.$ $\begin{pmatrix} err_u \\ err_v \end{pmatrix}$	$\begin{pmatrix} 53.816 \\ 56.798 \end{pmatrix}$	$\begin{pmatrix} 0.163 \\ 0.136 \end{pmatrix}$	$\begin{pmatrix} 0.159 \\ 0.142 \end{pmatrix}$

TAB. 4.11 – Étalonnage de la caméra munie d'une lentille Orifl 190-3 d'Omnitech Robotics. Cette caméra a un champ de vue de 190 degrés et une focale de 1.24 mm. La taille de l'image est fixée à 640×480 pixels.

($[0.163\ 0.136]^\top$ contre $[0.159\ 0.142]^\top$ pour la lentille Sigma) ou moins de précision dans le cas de la lentille Sigma ($[0.218\ 0.205]^\top$ contre $[0.153\ 0.168]^\top$) que le modèle unifié sans paramètres supplémentaires.

4.3.2.4 Reconstruction euclidienne partielle

La stratégie de suivi de chemin proposée nécessite l'estimation des variables d'état. Comme nous l'avons évoqué dans l'introduction de ce chapitre, ces variables peuvent être calculées à partir de la reconstruction euclidienne entre deux vues. Il est donc intéressant d'évaluer les performances de cette reconstruction lorsque le modèle unifié est employé. Nous utilisons pour cela des données synthétiques puis des données réelles.

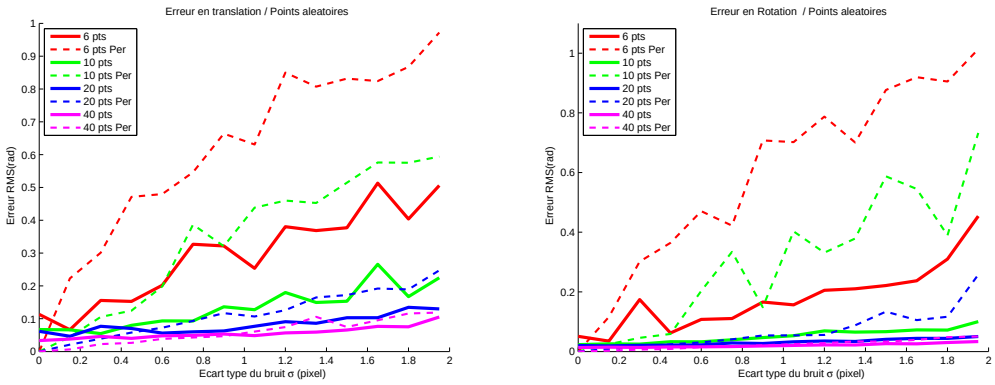
Évaluation de la reconstruction avec des données synthétiques Afin d'évaluer les performances de la reconstruction 3D partielle en utilisant le modèle unifié pour une caméra fisheye, la caméra est placée dans un espace de travail virtuel. Elle est dirigée vers un plan (π) contenant N points placés aléatoirement. Les images synthétiques de taille 640×480 pixels sont générées en projetant ces points sur le plan image à partir du modèle de projection $r_f^1(r_p)$ avec la valeur des paramètres de la caméra Orifl trouvés précédemment (*i.e* $r_f^1(r_p) = r_p(1 - 0.238.r_p^2 + 0.040.r_p^4 + 0.002.r_p^6 - 0.001.r_p^8)$). Un bruit gaussien 2D avec une moyenne nulle et un écart type σ est ajouté sur les coordonnées de chaque point image. Pour chaque expérience, deux positions aléatoires de la caméra sont générées avec un déplacement correspondant à une rotation $\mathbf{R}$ et une translation t. La matrice d'homographie $\mathbf{H}_\pi$ entre les deux images, relative au plan (π), est estimée à partir du modèle unifié de la caméra avec les paramètres de la caméra Orifl (*i.e* $\xi = 2.875$). La matrice de rotation $\widehat{\mathbf{R}}$ et la translation à un facteur d'échelle près $\widehat{\mathsf{t}}$ sont extraites de $\mathbf{H}_\pi$ [11] puis les erreurs suivantes sont calculées :

- erreur de rotation Θ_R : angle de rotation de la matrice $\mathbf{R}\widehat{\mathbf{R}}^{-1}$
- erreur de translation Θ_T : angle entre les vecteurs normalisés $\mathsf{t}/\|\mathsf{t}\|$ et $\widehat{\mathsf{t}}/\|\widehat{\mathsf{t}}\|$

Pour comparer les résultats avec ceux du cas connu des caméras perspectives (voir, par exemple, [Hartley 00]), la matrice d'homographie $\mathbf{H}_\pi$ est également estimée à partir du modèle de projection utilisé pour générer les images. Les résultats sont comparés avec la réalité terrain et l'erreur RMS (*Root Mean Square*) est calculée à partir de 200 tests pour chaque expérience. L'erreur RMS de la reconstruction est représentée en fonction de l'écart type du bruit de mesure avec le modèle de projection perspective (lignes pointillées) et avec le modèle unifié (lignes pleines), pour un mouvement de translation pure Fig. 4.14 (a) et un mouvement de rotation pure Fig. 4.14 (b).

On observe de manière classique que l'erreur RMS augmente quand l'écart type du bruit augmente et qu'elle diminue avec le nombre de points. En l'absence de bruit, le modèle perspective (utilisé pour générer les données) permet d'obtenir exactement le mouvement effectué, ce qui n'est pas le cas du modèle unifié. Cependant, le modèle unifié est plus robuste vis-à-vis des bruits de mesure surtout lorsque le nombre de

[11] Une estimation approximative de la normale au plan contenant les points 3D permet de lever l'ambiguïté de la décomposition.



(a) (b)

FIG. 4.14 – Reconstruction euclidienne partielle obtenue à partir de données synthétiques : erreur RMS (a) en translation Θ_T et (b) en rotation Θ_R à partir de 6, 10, 20 et 40 points en correspondance en fonction de l'écart type $\sigma = 0 \ldots 2$ pixel(s).

points utilisés pour estimer la matrice d'homographie est faible. De manière générale, les résultats de reconstruction euclidienne pour une caméra fisheye sous le modèle de projection unifié sont toujours similaires (voir plus satisfaisants) que les résultats sous le modèle sténopé.

Évaluation de la reconstruction avec des données réelles Trois déplacements de la caméra sont considérés : une translation $\mathtt{t}$, une rotation $\mathbf{R}$ et un mouvement composé ($\mathbf{R}$, $\mathtt{t}$). La réalité terrain est mesurée à partir de la position de la caméra sur une table à mesurer tridimensionnelle. La matrice d'homographie est estimée à partir de la position de quatre couples de points en correspondance dans les images acquises par l'objectif grand-angle Orifl190-3 aux positions initiale et finale. Les déplacements sont ensuite obtenus par décomposition de cette matrice et sont comparés à la réalité terrain. Les résultats sont donnés Tab. 4.12.

Les déplacements sont correctement estimés. Lors d'une translation pure, une erreur de translation inférieure à 1 degré est observée. Lors d'une rotation pure, l'erreur entre la rotation effectuée et la rotation estimée est d'environ 3 degrés. Quand les mouvements sont couplés, les erreurs sont inférieures à 2 degrés.

4.4 Conclusion

Dans ce chapitre, nous avons étudié les composantes de notre stratégie de navigation à mettre en place avec une caméra offrant un champ de vue important.

Tout d'abord, nous avons choisi d'utiliser des primitives de type points d'intérêt dans les trois étapes de cette stratégie. Ils ont pour avantage d'être utilisables dans

(a) Translation

$\mathsf{t}/\|\mathsf{t}\|$ (m)	0.843	-0.334	0.422
$\widehat{\mathsf{t}}/\|\widehat{\mathsf{t}}\|$ (m)	0.841	-0.344	0.417
Θ_T (deg.)	0.917		

(b) Rotation

$\mathsf{u}\theta$ (deg.)	0	-40.00	0
$\widehat{\mathsf{u}\theta}$ (deg.)	0.206	-39.4	-3.20
Θ_R (deg.)	3.20		

(c) Déplacement combiné

$\mathsf{t}/\|\mathsf{t}\|$ (m)	0.688	0	0.726
$\widehat{\mathsf{t}}/\|\widehat{\mathsf{t}}\|$ (m)	0.673	-0.015	0.740
Θ_T (deg.)	1.21		
$\mathsf{u}\theta$ (deg.)	0	-20.00	0
$\widehat{\mathsf{u}\theta}$ (deg.)	0.470	-21.77	-0.006
Θ_R (deg.)	1.831		

TAB. 4.12 – Reconstruction 3D partielle avec des données réelles. Calcul du déplacement : (a) en translation, (b) en rotation et (c) en rotation et translation. La rotation est représentée par $\mathsf{u}\theta$ (rotation d'axe u d'angle θ exprimé en degrés).

des environnements d'intérieur et d'extérieur. Toutefois, l'utilisation d'autres primitives peut être envisagée. Dans les environnements d'intérieur par exemple, de nombreuses droites 3D sont présentes et pourraient permettre une estimation plus robuste des variables d'état. On peut également envisager d'utiliser conjointement plusieurs types de primitives afin, une nouvelle fois, d'accroître la robustesse du système de navigation. Une autre possibilité est la définition d'un descripteur global permettant l'estimation du déplacement entre deux vues. Cela permettrait d'éviter la phase de détection nécessaire aux méthodes locales et donc de limiter le coût calculatoire. Cette problématique est largement ouverte et constitue une de nos perspectives.

Une méthode hiérarchique de localisation initiale a ensuite été proposée. Celle-ci est réalisée en deux temps. Dans un premier temps, un ensemble d'images candidates à la localisation sont extraites de la mémoire sensorielle en exploitant des descripteurs d'images globaux. Dans un second temps, des descripteurs locaux sont utilisés afin de sélectionner la meilleure image. Les résultats expérimentaux ont montré que cette méthode constituait un bon compromis entre précision, coût calculatoire et taille mémoire requise.

Pour la localisation initiale, nous avons employé une méthode classique de mise en correspondance d'images à partir de primitives géométriques. Il serait certainement fructueux d'exploiter d'autres types de méthodes, comme par exemple celles basées sur les "mots visuels" [Filliat 07, Angeli 08b] afin d'accélérer la phase de localisation initiale. Le concept de base de ces méthodes est que les descripteurs des primitives dans plusieurs images peuvent être très similaires. Un mot est donc associé à ces descripteurs similaires et est ajouté à un dictionnaire. À partir de ce dictionnaire, la localisation, apparentée à la recherche de mots dans un texte, peut être réalisée très rapidement. L'évaluation de ce type d'approche dans notre système de navigation est également une

de nos perspectives.

Enfin, nous avons exploité et validé le modèle de projection unifié pour les caméras fisheye. Ce type de caméras peut alors bénéficier directement des nombreux développements dédiés aux caméras catadioptriques centrales (traitement d'image, étalonnage, ...). De plus, les algorithmes d'estimation des géométries projective et euclidienne, conçus pour les caméras conventionnelles, peuvent être assez directement utilisés pour les capteurs catadioptriques centraux et, par conséquent, pour les caméras fisheye. En particulier, l'estimation des variables d'état intervenant dans nos stratégies de commande pourra être réalisée avec des images acquises par les caméras conventionnelles, catadioptriques centrales ou fisheye, sans modification de l'algorithme de reconstruction euclidienne.

Chapitre 5

Mise en œuvre et expérimentations

Pour évaluer un système global de navigation, de nombreuses expérimentations doivent être réalisées. Il s'agit à la fois de les effectuer dans des contextes différents (robots, capteurs et sites de navigation différents) mais également de les répéter. Bien entendu, la réalisation de ces expérimentations peut prendre du temps. Il faudrait, en effet, pouvoir tester tous les cas de figure et toutes les configurations possibles. En pratique, des contraintes matérielles (disponibilité du matériel, état de fonctionnement, lieux d'expérimentation, problèmes logiciels ...) et temporelles limitent souvent le nombre d'expérimentations menées. Nous verrons dans ce chapitre que nous avons tout de même pu effectuer un certain nombre d'expérimentations dans des conditions variées permettant ainsi de valider les principes de notre stratégie globale de navigation.

Les performances d'un système de navigation peuvent également être visualisées à travers certaines expérimentations type comme le bouclage et la navigation dans des environnements de grande taille. Les stratégies proposées se heurtent souvent à ce problème de passage à l'échelle. En effet, la réalisation d'une tâche de navigation dans des environnements de très grande taille (échelle d'une ville) est difficile à mettre en œuvre. Comme nous l'avons vu dans le chapitre 2, notre approche se prête en théorie bien à cela car la quantité d'information à mémoriser est relativement faible au regard des ressources actuellement disponibles sur PC. En pratique, cet objectif est, cependant, difficile à atteindre sans la mise en place d'un outil dédié à la gestion de la mémoire sensorielle.

Dans ce chapitre, nous présentons dans un premier temps l'environnement logiciel SOVIN (pour *SOftware for VIsual Navigation*) développé afin d'aborder les environnements de navigation de grande taille (voir Section 5.1). Les conditions expérimentales et les sites de navigation sont ensuite décrits dans les Sections 5.2 et 5.3. Finalement, notre approche est validée à travers des expérimentations sur différents types de robots et dans différents environnements dans les Sections 5.4 et 5.5.

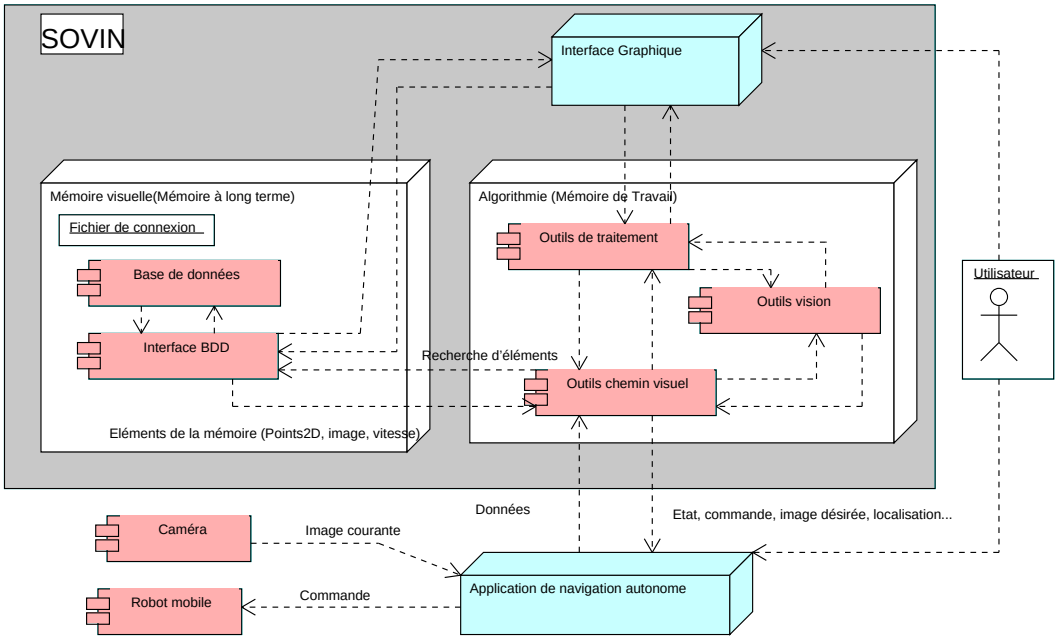


FIG. 5.1 – Implémentation de notre système complet de navigation.

5.1 Environnement logiciel

D'un point de vue expérimental, nous souhaitons aborder des sites de navigation de l'échelle d'une ville. Afin d'atteindre cet objectif, nous avons développé avec L. Lequièvre, ingénieur d'études CNRS au LASMEA, un logiciel nommé SOVIN dont les principes sont détaillés dans la suite.

5.1.1 Présentation brève du logiciel SOVIN

Le logiciel SOVIN se compose de trois modules (voir Figure 5.1). Le premier module est dédié à la gestion de la mémoire sensorielle. Une librairie permet de faire l'interface entre l'outil de gestion (décrit dans le paragraphe suivant) et les objets manipulés par les différents algorithmes. Le second module intègre les principaux algorithmes de traitements mathématiques et de vision (détection et appariement des primitives, reconstruction Euclidienne partielle ...). Enfin, le dernier module réalise l'Interface Homme Machine (IHM). Ce module permet de visualiser le contenu de la mémoire et de la modifier de manière interactive.

5.1.2 Gestion de la mémoire sensorielle

Un soin particulier a été porté au module gérant la mémoire sensorielle MS. Dans le contexte de la navigation de robots mobiles, le système de gestion de la carte de l'environnement doit répondre à plusieurs besoins :

	Élément de MS	Objet (O) ou relation (R) de la BDD
CS	Image $\mathcal{I}_i$	"Image" (O)
$CT1$	Noeud N_i Arête pondérée (N_i, N_j)	"Noeud" (O) "A_pour_noeud_suivant" (R)
$CT0$	Séquence Γ^i $CT0$	"Séquence" (O) "JeuTest" (O) "Contient_Séquence" (R)
CS - $CT1$	Capteur $\mathcal{I}_i \rightarrow N_i$	"Capteur" (O) "Contient_Image_Acquis_Par" (R)
$CT1$ - $CT0$	Noeud d'une séquence Arête entre séquences	"Contient_Noeud" (R) "A_pour_séquence_suivante" (R)
CS visuelle	Point image Relation image - point image Point3D physique Relation point image - point 3D	"Point2D" (O) "Contient_Point2D" (R) "Point3D" (O) "Est la projection de" (R)

TAB. 5.1 – Correspondance entre la structure de la mémoire sensorielle et les objets et relations de la BDD.

- les données doivent être sauvegardées, récupérées et gérées de façon efficace afin de supporter des environnements de grande taille,
- l'accès et la récupération des données doivent être rapides pour permettre une utilisation en temps réel,
- l'intégrité des données par rapport à la structure proposée doit être conservée,
- la structure et les données mémorisées doivent s'adapter à plusieurs approches de navigation.

Ces besoins ont guidé le développement d'un système de gestion de la mémoire sensorielle utilisant une Base De Données (BDD). Dans le contexte des BDD, de nombreux outils sont disponibles pour la conception, la gestion et la sauvegarde efficaces des données. En particulier, la méthode d'analyse et de conception MERISE a été employée pour décrire la structure correspondant à notre mémoire sensorielle. Cette méthode consiste en deux étapes : dans un premier temps, les entités de la BDD et leurs relations sont définies dans le Modèle Conceptuel de Données (MCD) ; dans un second temps, le Modèle Physique de Données (MPD) est généré à partir du MCD. Il est ensuite implémenté dans un moteur de base de données.

Les objets et leurs attributs puis les relations entre les objets sont directement définis dans le MCD à partir de la structure de la mémoire sensorielle MS présentée dans le Chapitre 2. Les correspondances entre les éléments de MS et ceux de la BDD sont décrites dans le Tableau 5.1. On retrouve les objets : "Image", "Noeud" et "Séquence" de MS. De plus, les séquences de $CT0$ acquises lors d'une même expérimentation peuvent être regroupées en "JeuTest". En pratique, plusieurs capteurs peuvent être employés pour les différentes phases d'apprentissage (objet "Capteur"). Des objets

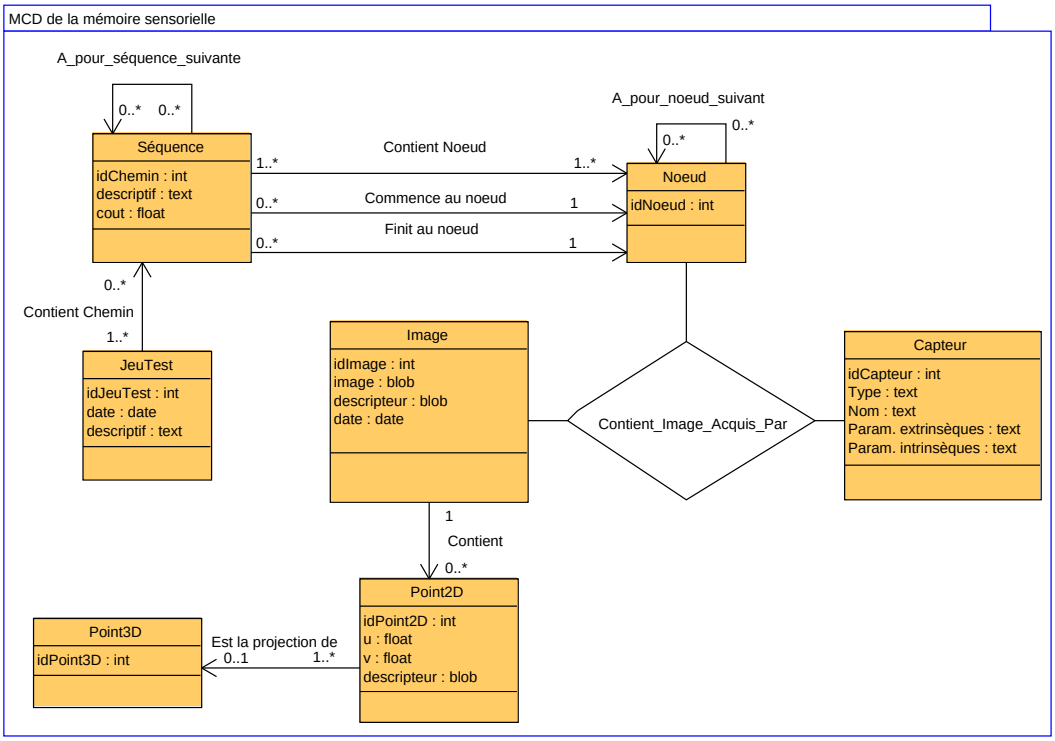


FIG. 5.2 – Éléments principaux du Modèle Conceptuel des Données de MS.

"Point2D" (correspondant à un point image) et "Point3D" (correspondant à un point physique de l'environnement)[1] sont également ajoutés au MCD.

Une "Image" a pour attribut le contenu de l'image (image en niveau de gris lorsqu'elle est acquise avec un capteur visuel) mais également le descripteur global. Un "Point2D" est défini par la position du point dans l'image ainsi que le descripteur du point. On note que la taille des images et des descripteurs n'est pas imposée. Les attributs d'un "Capteur" sont les paramètres intrinsèques et extrinsèques.

Des liaisons sont ensuite ajoutées au MCD. Ainsi, un JeuTest contient des Séquences (relation "Contient Séquence"). Une Séquence débute à un Noeud initial ("Commence_au_Noeud"), finit à un Noeud final ("Finit_au_noeud") et peut être suivie de plusieurs séquences (relation "A_pour_Séquence_Suivante"). Les arêtes de $CT1$ sont traduites dans le MCD par la relation "A_pour_Noeud_suivant". Noeud, Capteur et Image sont liés par la relation "Contient_Image_Acquis_Par". Les Point2D extraits dans une image sont liés à cette image par la relation "Contient_Point2D". Enfin, la relation "Est la projection de" lie un Point2D à un Point3D.

Les éléments principaux du MCD obtenu sont représentés Figure 5.2.

[1] Un point 3D peut être défini par ses coordonnées ou simplement traduire le fait que des points image sont appareillés.

Le Modèle Physique des Données, généré à partir du MCD, contient l'implémentation des structures physiques de la base de données. Dans ce modèle, chaque entité du MCD est exprimée dans une table et chaque attribut est représenté par une colonne de cette table. Un élément de cette entité correspond alors à une ligne de cette table, identifiée de manière unique par une *clé* (ou *identifiant*). Le MPD de la mémoire sensorielle contient une table pour chaque objet défini au paragraphe précédent (JEUTEST, SEQUENCE, CAPTEUR, NOEUD , IMAGE, POINT2D et POINT3D) et d'autres tables intermédiaires créées à partir des liaisons (CONTIENT_SEQUENCE, A_POUR_SEQUENCE_SUIVANTE, ARC et CONTIENT_IMAGE_ACQUIS_PAR). Les données relatives à ces tables sont accessibles à la modification et à la lecture via des requêtes qui utilisent les identifiants et les attributs des tables. Par exemple, une requête pour obtenir les identifiants (IDPOINT2D), les positions (U et V) et les descripteurs des points extraits d'une image (l'image d'identifiant 1 ici) utilise les données de la table POINT2D et s'écrit : `SELECT IDPOINT2D, U, V, DESC FROM POINT2D WHERE IDIMAGE=1`. Il est ainsi possible d'accéder rapidement aux différents éléments de la mémoire à partir de requêtes plus ou moins évoluées.

5.1.3 Mode opératoire

Les trois phases de notre stratégie de navigation sont réalisées en exploitant le logiciel SOVIN :

Étape 1 : la mémoire sensorielle est tout d'abord construite en trois étapes comme décrit dans la Section 2.2.1 :
- la première étape (sélection des images clés) est réalisée de façon automatique comme proposé dans la Section 4.1,
- la seconde étape (ajout des données dans la mémoire sensorielle) est automatique et consiste en l'insertion des données dans la BDD de SOVIN,
- la troisième étape (mise à jour de la carte) est réalisée manuellement via l'IHM.

Étape 2 : la phase de localisation initiale exploite l'approche décrite dans la Section 4.2.2. Afin d'accélérer les traitements, l'utilisateur peut désigner la séquence sur laquelle se situe le robot. La localisation est ensuite réalisée de façon complètement automatique. On peut noter que les primitives locales et les descripteurs globaux des images sont stockés dans la BDD et peuvent être récupérés séparément des images. Une fois l'étape de localisation terminée, l'utilisateur peut vérifier que le résultat est correct via l'IHM.

Étape 3 : la phase de navigation autonome (voir Section 2.2.3) est réalisée en deux temps :
- dans un premier temps, le chemin sensoriel Ψ est extrait de MS. Cette phase consiste à récupérer les identifiants des images de Ψ connaissant l'image initiale et l'image cible spécifiée par l'utilisateur,
- dans un second temps, le suivi de chemin est réalisé de façon automatique. Les images de Ψ et les points image associés sont tout d'abord récupérés

depuis la base de données via les identifiants des images[2]. Les points de l'image courante sont ensuite détectés et appareillés avec ceux de l'image cible. Le déplacement entre deux vues est obtenu comme décrit dans la Section 4.3.1.2 et permet d'estimer les variables d'état. L'utilisateur peut à tout moment stopper la phase automatique en cas d'anomalie. En outre, de nombreuses sécurités logicielles ont été intégrées (arrêt du robot si le nombre de points reconstruits n'est pas suffisant ou si les entrées de commande ont des valeurs aberrantes...).

5.2 Conditions expérimentales

Nous détaillons dans la suite le matériel utilisé lors de nos expérimentations.

5.2.1 Matériel informatique

Le logiciel SOVIN a été implémenté en C^{++} sur un PC utilisant un OS Linux RTAI avec un processeur Centrino de 2 GHz. La librairie Qt4 développée par Trolltech a été utilisée. Cette librairie, gratuite et multiplateforme, à l'avantage d'offrir des fonctionnalités pour la communication avec les bases de données. Le système de gestion de la base de données représentant la mémoire sensorielle est MySQL.

5.2.2 Robots mobiles

Trois robots mobiles ont été utilisés lors de nos expérimentations : un robot mobile à roues Pioneer, un véhicule urbain nommé RobuCab et un quadrirotor développé au CEA-LIST.

Robot Pioneer Ce robot tout-terrain électrique Pioneer 3-AT est commercialisé par ActivMedia (voir Fig. 5.3). Il est bien adapté à la navigation en intérieur de par ses dimensions (50 centimètres de long, 49 centimètres de large, 26 centimètres de haut) mais peut également être utilisé en environnement extérieur. Les roues, non directrices, sont actionnées par des moteurs électriques alimentés par une batterie permettant une autonomie de 3 à 6 heures. Chaque moteur est commandé par un contrôleur bas-niveau et est muni d'un encodeur. La librairie ARIA (*Advanced Robot Interface for Applications*), développée par MobileRobots, permet de communiquer avec le robot et ses composants via une liaison série.

Ce robot est décrit par le modèle cinématique char. L'objectif de commande est réalisé par la Stratégie 1 (voir Section 3.2.1) dans laquelle le Critère 2 de changement d'image est employé (voir Section 3.3.1.1).

[2]Les images ne sont récupérées que pour l'affichage.

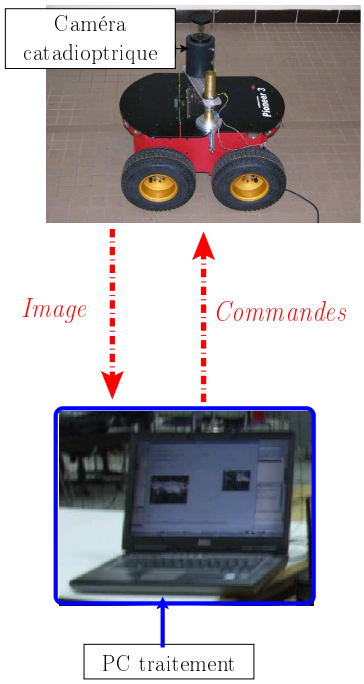


FIG. 5.3 – Robot Pioneer et équipements.

RobuCab Dans le cadre de la navigation en milieu urbain, nous utilisons un véhicule appelé RobuCab, industrialisé par la compagnie Robosoft (voir Fig. 5.4). Ce véhicule est actionné par un moteur électrique sur chaque roue, alimenté par des batteries plomb-acide pour une autonomie de 2 heures. Il est spécifiquement conçu pour les zones où la circulation automobile est fortement restreinte : centre-ville urbain, gare, aérogare, site touristique, hôpital, campus universitaire ... Ses dimensions peu importantes (1.90 mètre de long, 1.20 mètre de large, 2.20 mètres de haut) sont, en effet, bien adaptées à de tels environnements. Un ordinateur embarqué permet de gérer la commande manuelle via un joystick et l'accès aux commandes de bas niveau. Notre PC et l'ordinateur du RobuCab sont connectés via un bus CAN. L'interface entre les données fournies par les capteurs, la commande et notre système de navigation est réalisée via l'architecture multi-capteurs temps réel Aroccam développée conjointement au Cemagref de Clermont-Ferrand et au LASMEA (voir [Tessier 06]).

Le véhicule est équipé d'un DGPS différentiel cinématique temps réel RTK-DGPS Sagitta de Thales qui a une précision de 1 centimètre d'écart type dans le plan horizontal et de 20 centimètres sur l'axe vertical. Afin d'évaluer les performances de notre approche, les données acquises par ce DGPS seront utilisées comme réalité terrain. Les données acquises lors du suivi automatique seront comparées à celles acquises lors de la phase d'apprentissage et permettront d'estimer l'écart latéral entre les trajectoires parcourues.

Dans nos expérimentations, nous emploierons la configuration du RobuCab {roues avants orientables, roues arrières non orientables}. La configuration cinématique de

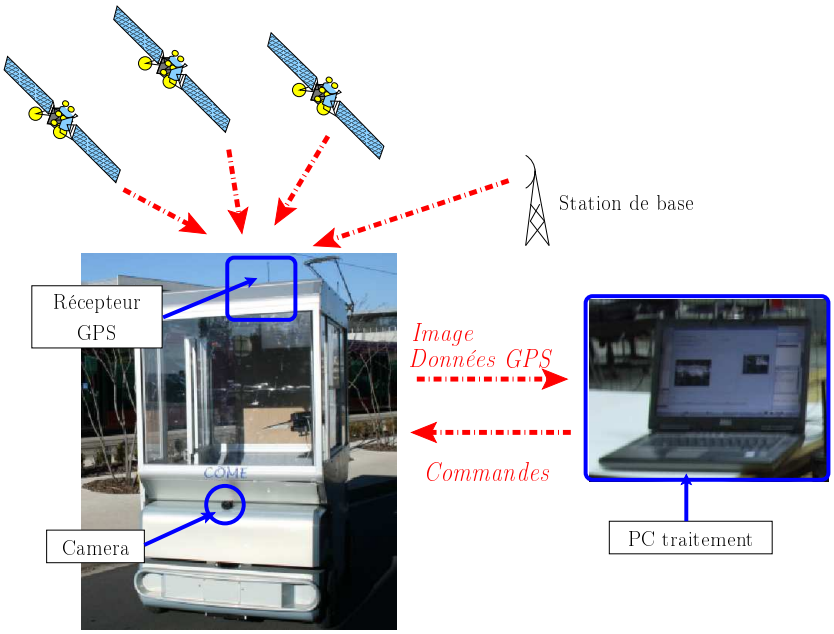


FIG. 5.4 – RobuCab et équipements.

ce véhicule suit alors le modèle bicyclette. Par manque de temps, nous n'avons pas pu mettre en œuvre la Stratégie 2. L'objectif de commande sera donc réalisé par la Stratégie 1. Le Critère 2 de changement d'image sera employé (voir Section 3.3.1.1). Afin d'obtenir un comportement confortable pour les passagers et sûr pour les actionneurs, la distance d'établissement est fixée à $d_m = 15$ mètres pour ce véhicule.

Drone quadrirotor Le drone quadrirotor développé au CEA-List est actionné par 4 moteurs électriques *brushless* alimentés par une batterie Lithium-Polymère. Il mesure 80 centimètres de large et de long. Son autonomie en vol atteint 20 minutes en régime quasi stationnaire. Il pèse environ 700 grammes et a une charge utile de 200 grammes, permettant ainsi d'embarquer une ou deux caméras et un capteur supplémentaire. Pour simplifier le développement de l'électronique embarquée, le drone est équipé de différentes cartes électroniques empilables (voir Fig. 5.5). La première carte (Carte moteurs) gère l'asservissement des vitesses de rotation des moteurs et génère les différentes tensions d'alimentation nécessaires aux cartes électroniques à partir de la tension batterie. La deuxième carte embarque une Centrale Inertielle (CI) composée de 3 accéléromètres, 3 gyromètres et 2 magnétomètres de technologie MEMS (*Micro Electro Mechanical System*). Une troisième carte est dédiée à la gestion des capteurs proximétriques (baromètre, ultrason, infrarouge...). Le DSP (*Digital Signal Processor*) effectue les opérations de filtrage et fusion des données de la centrale et implémente les algorithmes de stabilisation du drone, à une cadence de 6 millisecondes.

La communication entre le drone et le poste au sol est réalisée par une communication sans fil utilisant le protocole ZigBee (voir Fig. 5.6). Les déplacements du drone

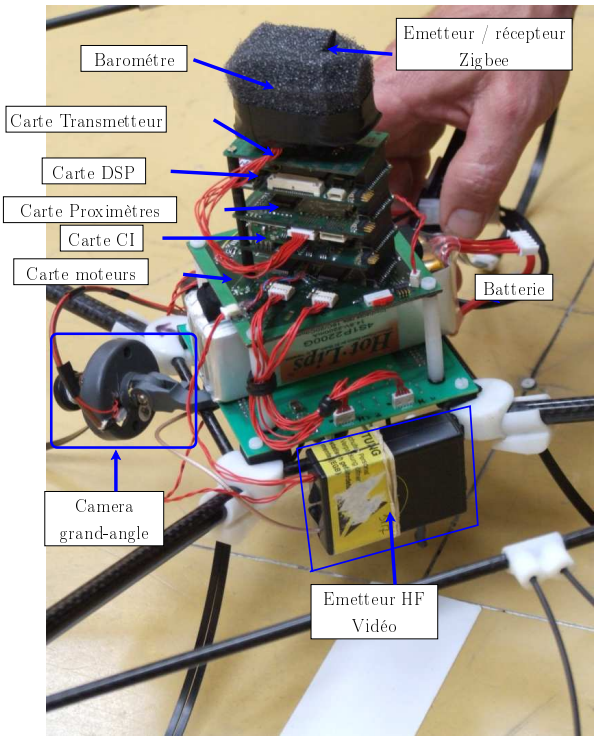


FIG. 5.5 – Cartes électroniques embarquées sur le quadrirotor.

peuvent être commandés depuis ce poste au sol à l'aide d'un joystick. Les algorithmes développés sur ce poste (interface et communication avec le drone) sont implémentés en C/C^{++} sous Windows. Un convertisseur analogique / numérique GrabBeeX transforme le signal vidéo émis depuis le drone en un signal numérique utilisable par le PC. Les images de mauvaise qualité sont éliminées.

L'attitude du drone est estimée à partir des données des accéléromètres et des gyromètres à l'aide d'un filtre complémentaire [Hamel 06b] tandis que la stabilisation en assiette de l'engin est effectuée avec les algorithmes développés dans [Guénard 04]. La stabilisation en vitesse nécessite l'estimation de la vitesse de translation v du drone. En pratique, nous avons utilisé deux méthodes pour estimer v. Celles-ci emploient une caméra embarquée sur le drone, dirigée vers le sol :

Méthode 1 : la vitesse est estimée à partir des données de la centrale inertielle et est recalée grâce à la mesure du flot optique sur le sol en utilisant une approche par logique floue. Une demande de brevet a été déposée au CEA-List par N. Guénard, ingénieur au CEA-LIST, concernant cette méthode.

Méthode 2 : la vitesse est estimée à partir d'un appariement de l'image courante avec une mosaïque construite en ligne et en utilisant une fusion avec les données de la centrale inertielle (approche développée par L. Eck, ingénieur au CEA-LIST, dans le cadre du projet européen μDrones).

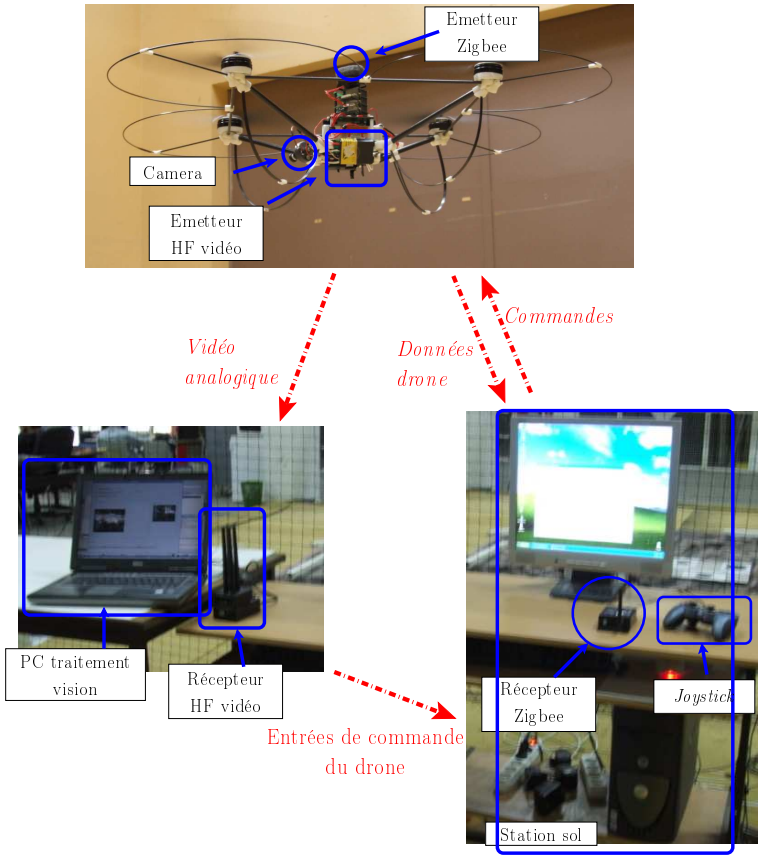


FIG. 5.6 – Drone quadrirotor et équipements.

Ces méthodes ne permettant pas d'estimer correctement la vitesse de translation suivant Z_c, nous ne traiterons que de la commande en position dans le plan $\mathsf{X}_c\mathsf{Y}_c$ et de la commande en lacet, la commande en altitude étant réalisée manuellement. Nous utiliserons alors le critère de changement d'image clé défini comme la norme de l'erreur de position dans le plan : $err = \|[ErrX\ ErrY]^T\|$ avec un seuil fixé à 0.05 mètre.

5.2.3 Capteurs visuels

Les trois capteurs visuels utilisés sont décrits par le modèle unifié et ont été étalonnés avec la toolbox développée par C. Mei *et al.* [Mei 07]. Dans chaque cas, les paramètres extrinsèques ont été estimés grossièrement.

Caméra catadioptrique analogique Ce capteur est composé d'un miroir parabolique couplé avec une caméra orthographique. Les images de résolution 640×480 pixels, converties en signaux numériques via un convertisseur GrabBeeX, sont acquises en niveau de gris à une fréquence de 7.5 Hz. Les paramètres du modèle de caméra sont : $f_u = 176.3$,

FIG. 5.7 – Véhicule électrique RobuCab muni de la caméra fisheye Fujinon, dirigée vers l'avant du véhicule.

$f_v = 160.0$, $u_0 = 333.8$ pixels, $v_0 = 281.8$ pixels et $\xi = 1$. Cette caméra est fixée sur la base du Pioneer et son axe optique est approximativement confondu avec l'axe de rotation du robot (voir Fig. 5.3).

Caméra fisheye Fujinon Ce système visuel est constitué d'une caméra CMOS et d'un objectif grand-angle Fujinon Fisheye E5 1 :1.4/1.4 mm dont le champ de vue est de 185 degrés. Dans nos expérimentations, les images, de résolution 800×600 pixels, sont acquises en niveau de gris sur le PC à une fréquence de 15 Hz. Les paramètres du modèle de caméra sont : $f_u = 264.9$, $f_v = 263.8$, $u_0 = 400.4$ pixels, $v_0 = 304.8$ pixels et $\xi = 1.63$. Cette caméra est approximativement placée à l'avant du RobuCab, sur la gauche et à 1 mètre du sol. Son axe optique est dirigé vers l'avant du véhicule (voir Fig. 5.7).

Caméra Drone Pour les applications sur le drone, les contraintes de dimension et de poids ont conduit au choix d'un système visuel léger. La caméra est munie d'une lentille de focale 2.1 mm permettant d'obtenir un champ de vue de 110 degrés. La masse de l'ensemble { caméra + objectif } est alors inférieure à 13 grammes. Les images, de résolution 640×480 pixels sont converties en niveau de gris. Les paramètres du modèle de caméra sont : $f_u = 429.5$, $f_v = 467.1$, $u_0 = 345.4$ pixels, $v_0 = 292.4$ pixels et $\xi = 2.52$. Lors des expérimentations, cette caméra est placée sous les cartes du drone et est dirigée dans la direction X_c (voir Fig. 5.6).

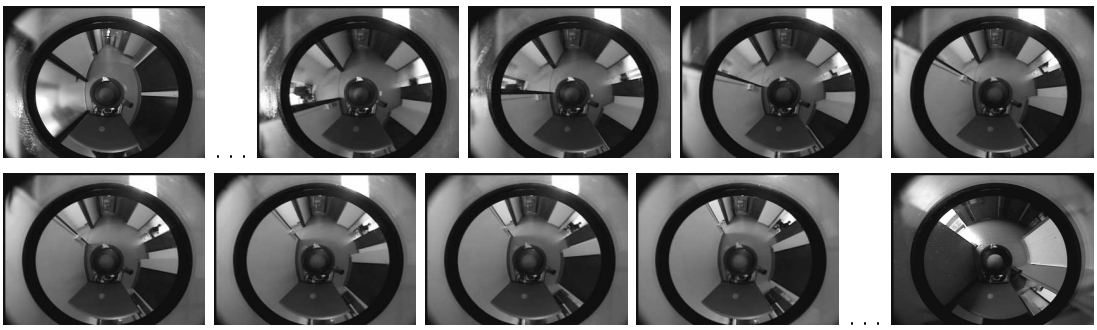


FIG. 5.8 – Quelques images de la séquence *Couloir*.

5.3 Sites de navigation

Nos expérimentations ont été conduites sur différents sites. Nous les présentons ici à travers les phases d'apprentissage réalisées.

5.3.1 Phases d'apprentissage avec le Pioneer

Séquence *Couloir* Le robot Pioneer est commandé manuellement dans un couloir le long d'une trajectoire de 12 mètres en ligne droite, avec une vitesse d'avance de 200 mm/s. 370 images sont acquises par la caméra catadioptrique embarquée. La construction de la carte conduit à une séquence unique (*Couloir*) contenant 75 images clés. La distance moyenne entre les positions d'acquisition de ces images est de 15 centimètres. Quelques images clés de la séquence *Couloir* sont représentées Fig. 5.8. On observe sur ces images des conditions d'illumination différentes selon la situation du robot. De plus, comme il est visible, par exemple, sur la première image représentée Fig. 5.8, ce type d'environnement contient de nombreuses zones non texturées où il sera difficile de détecter des points d'intérêt.

5.3.2 Phases d'apprentissage avec le RobuCab

Grand Environnement* et *Chemin simple Le RobuCab a été conduit manuellement le long de plusieurs trajectoires sur le Campus des Cézeaux (voir Fig. 5.9 [3]). Les trajectoires parcourues contiennent des virages serrés (près des lieux *STAPS*, *IFMA* et le virage en U au lieu *Tramway*) ainsi que des montées et descentes (entre *ISIMA* et *IFMA* puis vers *IFMA*). La mémoire sensorielle relative à ce site de navigation (appelée *Grand Environnement* dans la suite) contient 47 séquences et 1400 images clés. Toutes ces séquences ont été enregistrées avec un temps ensoleillé. Lorsque le soleil est bas et dans le champ de vue du capteur, celui-ci est saturé ce qui rend inutilisable une partie de l'image comme on le voit sur l'image représentée en bas à droite Fig. 5.9. Comme

[3] Les données DGPS sont utilisées afin de représenter le chemin dans un repère métrique absolu. L'image satellite de fond a été recalée manuellement sur ces données.

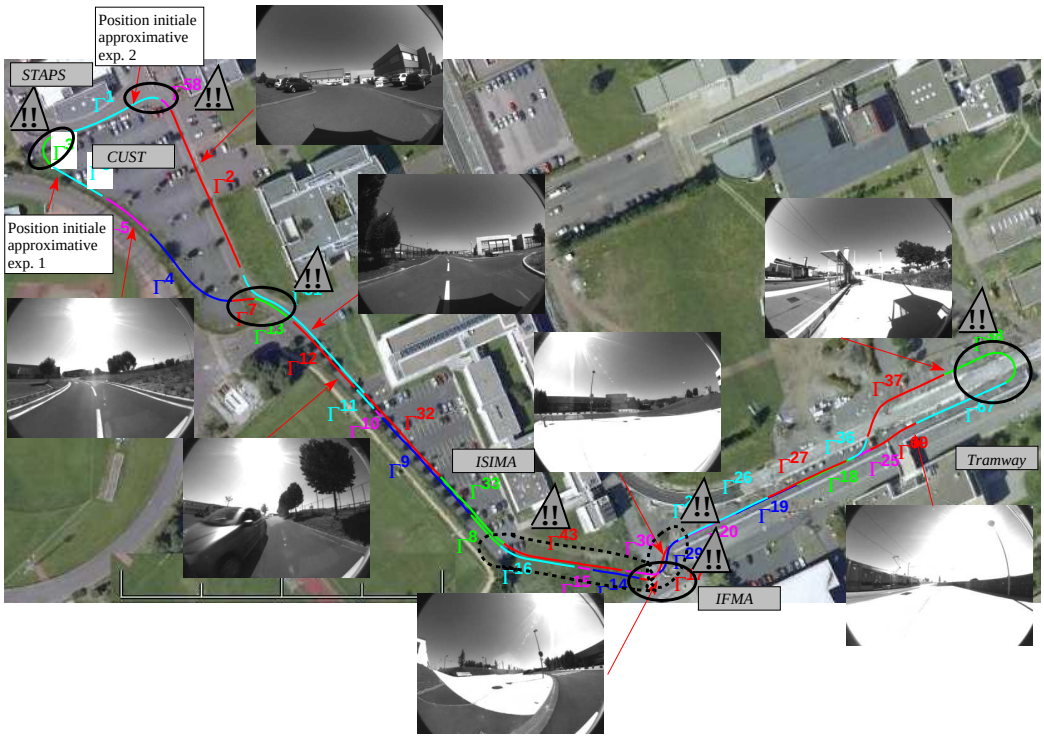


FIG. 5.9 – Trajectoires parcourues lors de la construction de la mémoire sensorielle *Grand Environnement*.

le champ de vue du capteur est important, il reste, cependant, suffisamment de primitives détectées dans le reste de l'image. Des phénomènes de réverbération sur les sols clairs apparaissent en raison de la position du soleil comme on peut le voir sur les trois images représentées près du lieu *Tramway*. Nous verrons dans la suite que la tâche de navigation autonome peut tout de même être menée à bien dans ces conditions.

Nous considérons également une sous partie de la mémoire *Grand Environnement* correspondant à la trajectoire de 200 mètres représentée Figure 5.10. Cette séquence, appelée *Chemin simple* dans la suite, contient 110 images clés.

Boucle "CUST" Lors d'une nouvelle phase d'apprentissage, un trajet en boucle long de 270 mètres est parcouru (*Boucle "CUST"*). 1100 images sont acquises par la caméra embarquée. La mémoire sensorielle est constituée de trois séquences et contient un total de 153 images clés (soit une moyenne d'une image tous les 1.7 mètres). Les positions du robot correspondant aux prises de vue de ces images sont représentées Figure 5.11.

5.3.3 Phases d'apprentissage avec le drone

Les expérimentations menées sur le drone sont réalisées dans un environnement d'intérieur de type hangar. Deux séquences sont considérées.

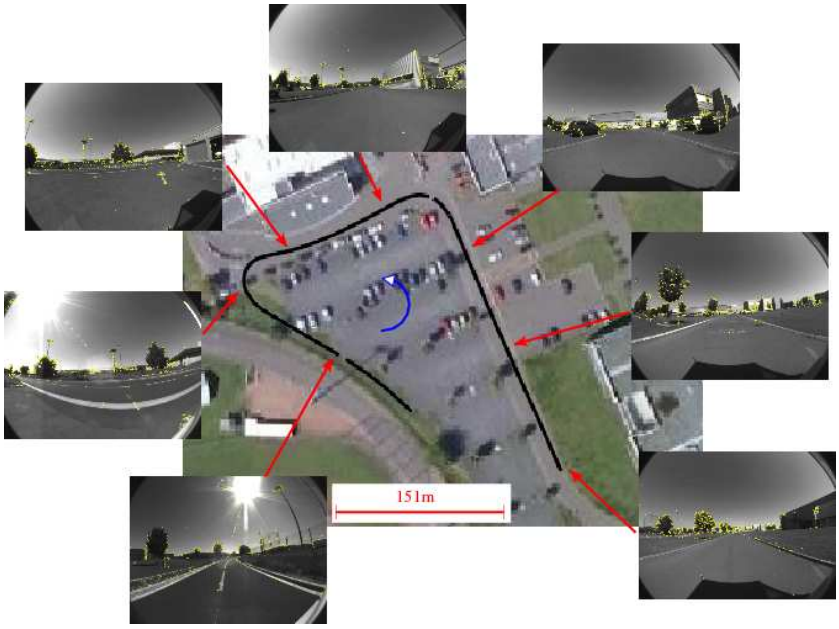
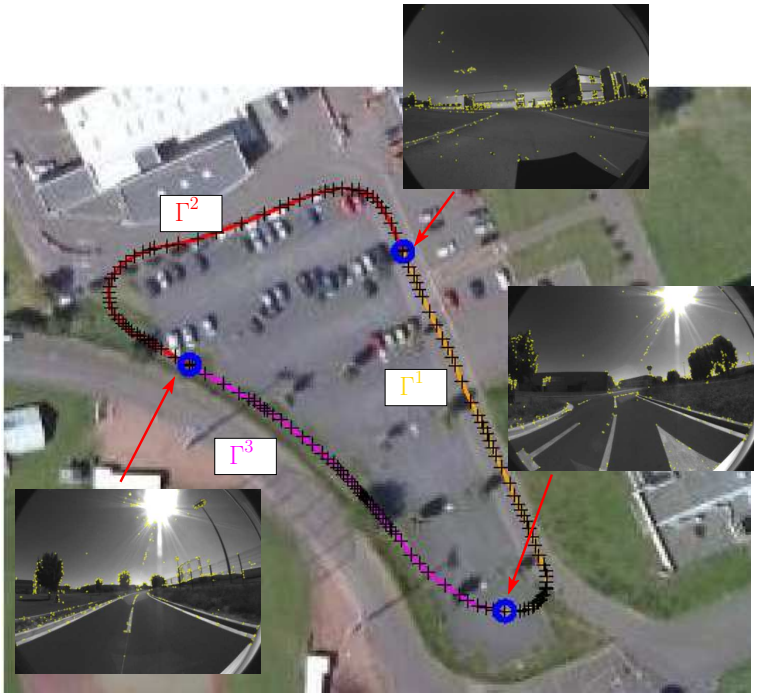


FIG. 5.10 – Trajectoire correspondant aux lieux d'acquisition des images de la séquence *Chemin simple*.

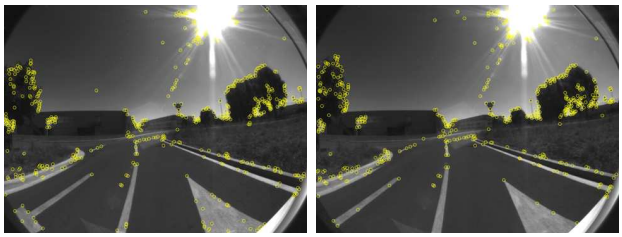
Séquence *Drone I* Le drone est commandé manuellement le long d'un chemin horizontal, approximativement linéaire, dirigé à 45 degrés par rapport à la direction d'avance X_c du drone et long de 6 mètres environ (voir Fig. 5.12). La sélection des images clés est, dans le cas du drone, réalisée manuellement en raison de la mauvaise qualité de certaines images des séquences d'apprentissage. La mémoire visuelle obtenue contient 10 images clés (voir Fig. 5.12).

Séquence *Drone II* Cette séquence a été acquise le long d'un chemin d'apprentissage en ligne droite dans la direction d'avance X_c du drone. Le chemin total mesure plus de 15 mètres. Après sélection manuelle des images clés, la mémoire visuelle est formée d'une séquence contenant 11 images (voir Fig. 5.13).

Pour ce type de robot, les graphes $CT1$ et $CT0$ sont non-orientés. On suppose, en effet, que si le robot peut se déplacer du lieu d'acquisition de $\mathcal{I}_i$ à celui de $\mathcal{I}_{i+1}$, alors il peut également se déplacer du lieu d'acquisition de $\mathcal{I}_{i+1}$ à celui de $\mathcal{I}_i$. Nous verrons dans l'expérimentation décrite Section 5.4.3.1 que cette hypothèse est tout à fait raisonnable.



(a)



(b) (c)

FIG. 5.11 – *Boucle "CUST"*. (a) chemin, représentée en coordonnées métriques, effectué lors de la phase d'apprentissage, (b) dernière image de Γ^3, (c) première image de Γ^1.

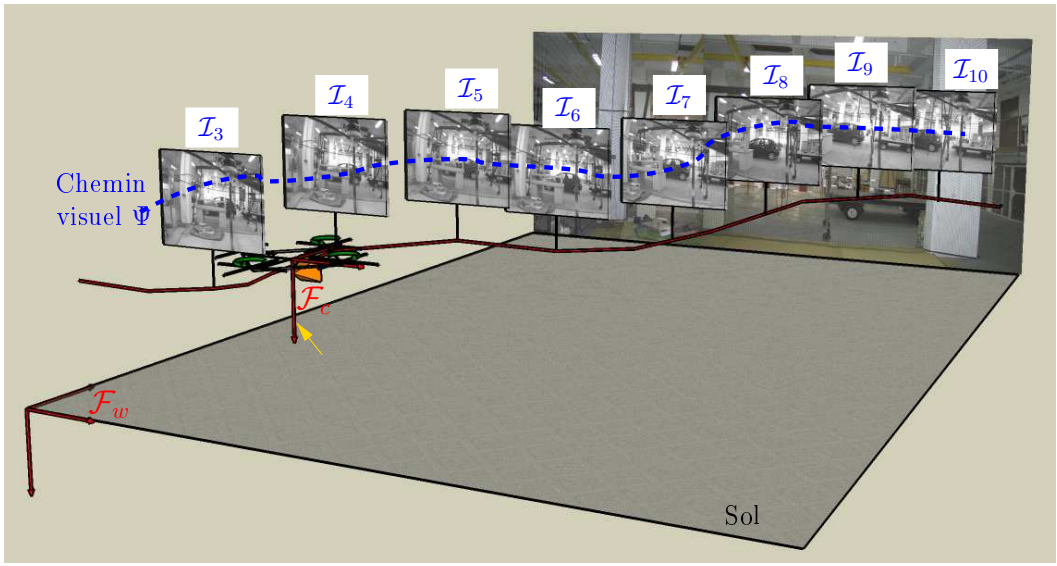
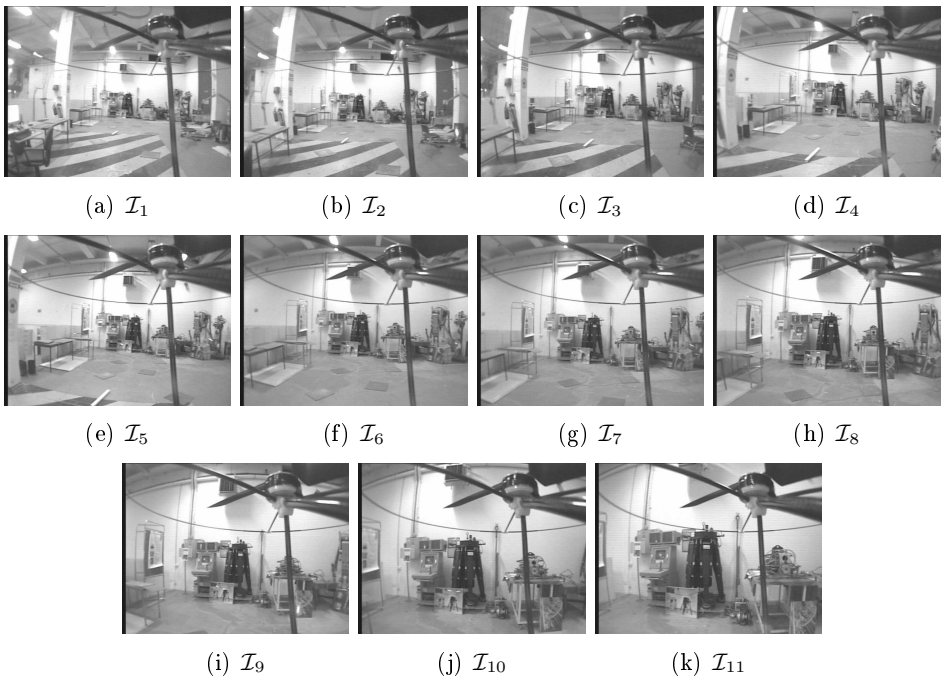


FIG. 5.12 – Représentation schématique du chemin parcouru lors de la phase d'apprentissage de la séquence *Drone I* avec quelques unes des images clés.



(a) $\mathcal{I}_1$ (b) $\mathcal{I}_2$ (c) $\mathcal{I}_3$ (d) $\mathcal{I}_4$

(e) $\mathcal{I}_5$ (f) $\mathcal{I}_6$ (g) $\mathcal{I}_7$ (h) $\mathcal{I}_8$

(i) $\mathcal{I}_9$ (j) $\mathcal{I}_{10}$ (k) $\mathcal{I}_{11}$

FIG. 5.13 – Images de la séquence *Drone II*.

5.4 Validation

L'objectif de cette partie est de valider notre stratégie de navigation au travers de diverses expérimentations. Celles-ci sont menées sur les trois robots considérés, avec des caméras différentes (caméra fisheye et caméra catadioptrique) et dans des environnements d'intérieur et d'extérieur.

5.4.1 Navigation en intérieur avec une caméra catadioptrique

Le Pioneer est positionné près du lieu d'acquisition de la séquence *Couloir*. Il a pour objectif de rejoindre le lieu d'acquisition de l'image finale de cette séquence. Le robot est tout d'abord localisé automatiquement comme proposé dans le Chapitre 2. Un chemin est ensuite extrait entre l'image initiale et la configuration désirée puis la phase de suivi débute avec une vitesse longitudinale de 100 mm/s. 35 couples de points sont mis en correspondance de façon robuste entre l'image courante et la première image du chemin sensoriel (voir Fig. 5.14 (a)). L'erreur latérale initiale est $y(s_0) \approx 1.5$ mètre [4].

Le chemin sensoriel est entièrement suivi. Les erreurs angulaire et latérale et la commande sont représentées Figure 5.15[5]. Le robot rejoint le chemin appris au bout de 5 secondes. Ensuite, la valeur absolue de l'erreur angulaire est inférieure à 2 degrés, celle de l'erreur latérale inférieure à 30 centimètres et celle de la commande inférieure à 1 deg/s (voir Fig. 5.15). Lors du suivi, l'erreur dans l'image est inférieure à 15 pixels et atteint, dans la majorité des cas, une valeur d'environ 4 pixels lorsque l'image clé est rejointe (voir Fig. 5.16). Lors de cette expérimentation, 49 appariements robustes sont obtenus en moyenne et permettent d'estimer les entrées de commande.

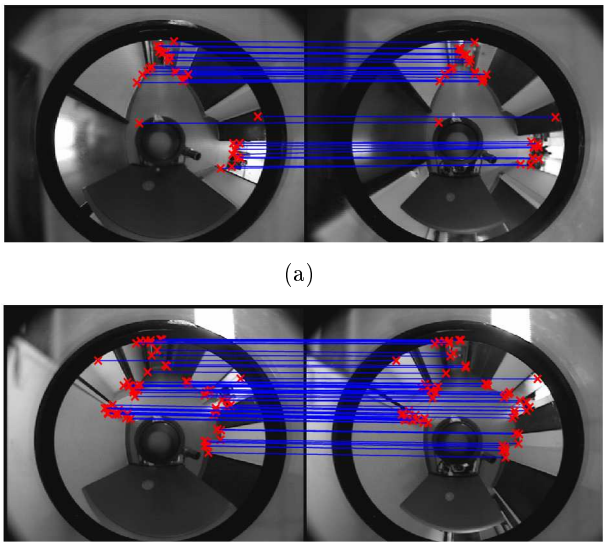
À travers cette expérimentation, nous voyons que notre stratégie peut être appliquée avec une caméra catadioptrique. Une telle caméra couvrant un champ de vue plus important qu'une caméra classique, des points d'intérêts sont détectés tout autour du robot comme on peut le voir sur les images représentées Fig. 5.14. Dans une telle application, on note également que la présence de la zone morte au centre de l'image ne constitue pas un inconvénient majeur car celle-ci correspond à la projection du robot.

5.4.2 Navigation en milieu urbain

Nous évaluons ici les performances de notre stratégie dans un environnement d'extérieur avec le RobuCab équipé d'une caméra fisheye. Celui-ci est tout d'abord conduit manuellement vers une position proche de celle correspondant au début de la séquence *Chemin simple*. Après la phase de localisation initiale, un chemin visuel composé de 94 images est extrait de la mémoire sensorielle afin de rejoindre la configuration désirée (correspondant à l'image finale du chemin appris). La vitesse d'avance est fixée à V = 0.9 m/s.

[4] Étant données les caractéristiques de notre environnement, le facteur d'échelle est ici surestimé.

[5] Sur ces figures, les croix marquent un changement d'image clé.



(a)

(b)

FIG. 5.14 – Images acquises par la caméra lors de la phase automatique (à gauche) et images de la mémoire correspondantes. Les segments relient les primitives visuelles appareillées de façon robuste entre ces images. (a) Image initiale, (b) image en cours de navigation.

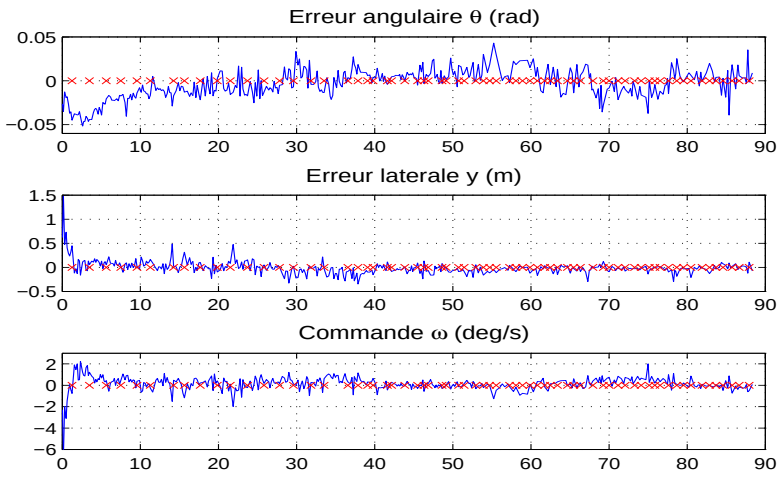


FIG. 5.15 – Séquence *Couloir* : erreur angulaire θ (exprimée en radians), erreur latérale y (exprimée en mètres) et commande (exprimée en deg/s) en fonction du temps (en secondes).

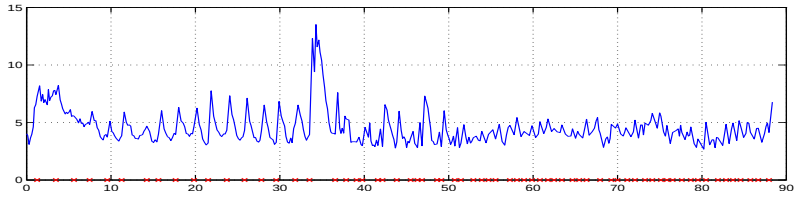


FIG. 5.16 – Séquence *Couloir* : erreur dans l'image ErrImage (exprimée en pixels) en fonction du temps (exprimé en secondes).

L'image désirée est atteinte par le RobuCab après 165 secondes de navigation autonome. Il a parcouru environ 150 mètres et plus de 2 000 images ont été traitées. Les chemins parcourus lors des phases d'apprentissage et de navigation autonome sont représentés Fig. 5.17. La trajectoire réalisée lors de l'apprentissage est suivie avec un écart latéral moyen de 20 centimètres et d'écart type 15 centimètres. Les erreurs angulaire et latérale ainsi que la commande sont représentées Fig. 5.18. On observe que l'erreur angulaire n'est pas régulée à zéro entre les temps 40 et 50 secondes et entre les temps 90 et 110 secondes. Lors de ces périodes, le robot était situé dans les virages serrés. La distance d'établissement étant élevée (15 mètres), la Stratégie 1 ne permet pas de totalement réguler à zéro les erreurs dans les virages, comme nous l'avons vu en simulation.

Un écart important par rapport à la trajectoire de référence, comme dans la situation présente, peut être problématique du point de vue de la perception. La mise en correspondance des points extraits de l'image courante avec ceux de l'image désirée s'avère alors difficile (scores de corrélation trop faibles) voire impossible (sortie du champ de vue).

La Figure 5.19 montre une séquence de 14 images successives acquises lors de la phase de navigation autonome entre deux images clés $\mathcal{I}_i$ et $\mathcal{I}_{i+1}$ se situant dans le Virage 2. Cette figure permet de visualiser les erreurs dans l'image, les segments en rouge liant les points de l'image courante aux points correspondants dans l'image désirée $\mathcal{I}_{i+1}$. On observe qu'à la fin de la séquence, l'erreur est quasiment nulle. On peut également noter que la répartition spatiale des points n'est pas homogène. Cela peut s'expliquer par le contenu peu texturé de l'image, principalement occupée par le macadam et le ciel. Il pourrait être fructueux de détecter les zones peu texturées. Cela permettrait de cibler les zones les plus intéressantes en terme de contenu pour le traitement d'image, diminuant ainsi son coût calculatoire.

5.4.3 Navigation du drone

Nous nous intéressons ici à la navigation autonome du drone le long des séquences *Drone I* et *Drone II*. Contrairement aux robots à roues étudiés précédemment, cet engin se déplace en trois dimensions, a une dynamique importante et est très sensible

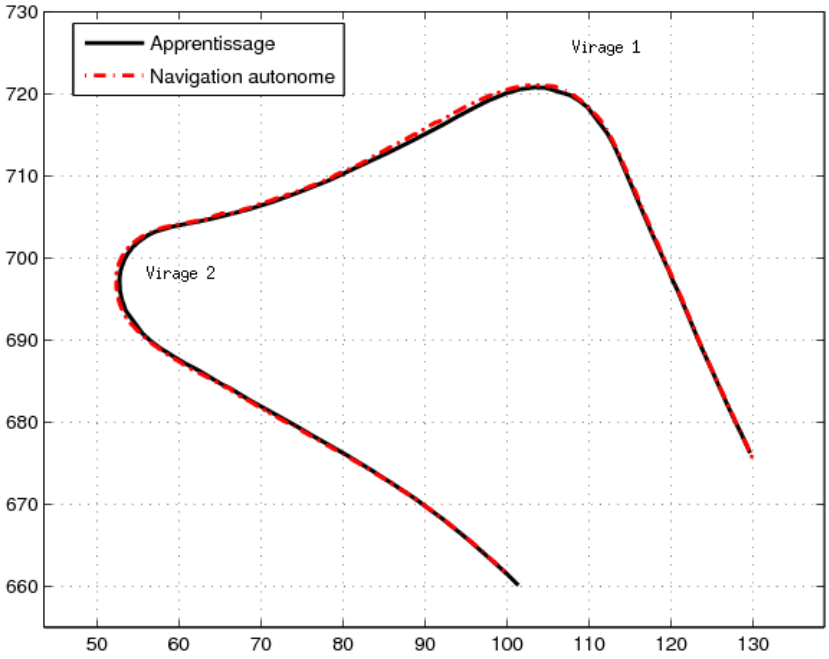


FIG. 5.17 – Séquence *Chemin simple* : chemins (exprimés en coordonnées métriques) parcourus lors des phases d'apprentissage et de navigation automatique.

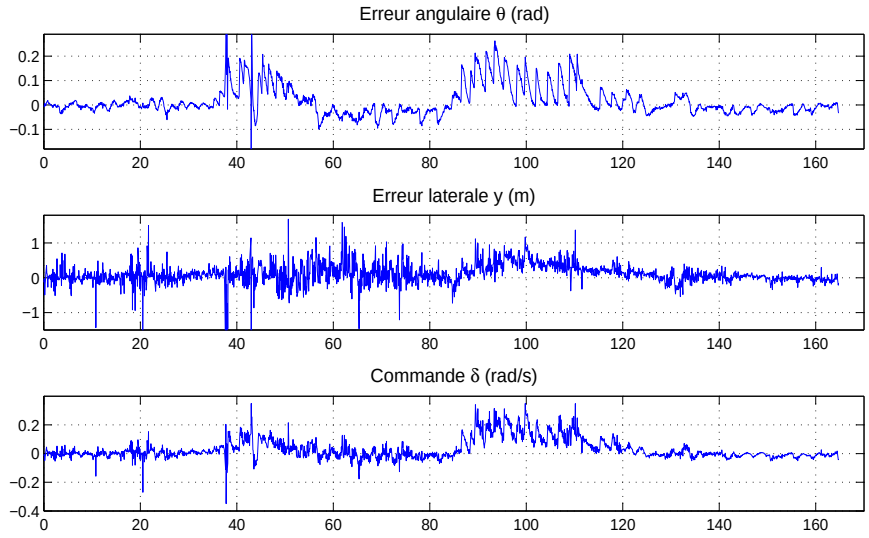


FIG. 5.18 – Séquence *Chemin simple* : erreur angulaire θ (exprimée en radians), erreur latérale y (exprimée en mètres) et commande δ (exprimée en deg/s) en fonction du temps (en secondes).

aux perturbations extérieures.

5.4.3.1 Séquence *Drone I*

Le quadrirotor est téléopéré vers une position proche de celle correspondant au lieu d'acquisition de l'image $\mathcal{I}_3$ de la séquence *Drone I*. Le chemin visuel est défini manuellement comme $\Psi = \{\mathcal{I}_3, \mathcal{I}_4, \ldots \mathcal{I}_{10}, \mathcal{I}_9, \ldots \mathcal{I}_2, \mathcal{I}_3, \ldots \mathcal{I}_{10}, \mathcal{I}_9, \ldots \mathcal{I}_2\}$. Nous utilisons la Méthode 1 pour l'estimation de la vitesse de translation (voir Section 5.2.2). Dans cette expérimentation, nous désirons observer le comportement en translation du drone. La commande en lacet n'est donc pas appliquée.

Lors de navigation autonome, le drone se déplace jusqu'à atteindre la dernière image du chemin sensoriel. Lors de cette expérimentation qui a duré 200 secondes, 50 couples de points ont été appareillés en moyenne. Les erreurs $ErrX$ et $ErrY$ sont représentées Fig. 5.20. Ces erreurs atteignent des valeurs proches de zéro à chaque image clé. Au début du suivi, on observe des oscillations de l'erreur de position avant d'atteindre les images clés (principalement sur l'axe X_c). Nous pensons que ces oscillations sont dues en partie à une mauvaise estimation de la vitesse de translation et plus particulièrement à une mesure erronée du flot optique. Deux principaux facteurs peuvent être mis en cause : le manque de texture sur la zone observée et la mauvaise qualité des images reçues par le convertisseur GrabBeeX.

Nous vérifions également que les graphes $CT1$ et $CT0$ peuvent être choisis non

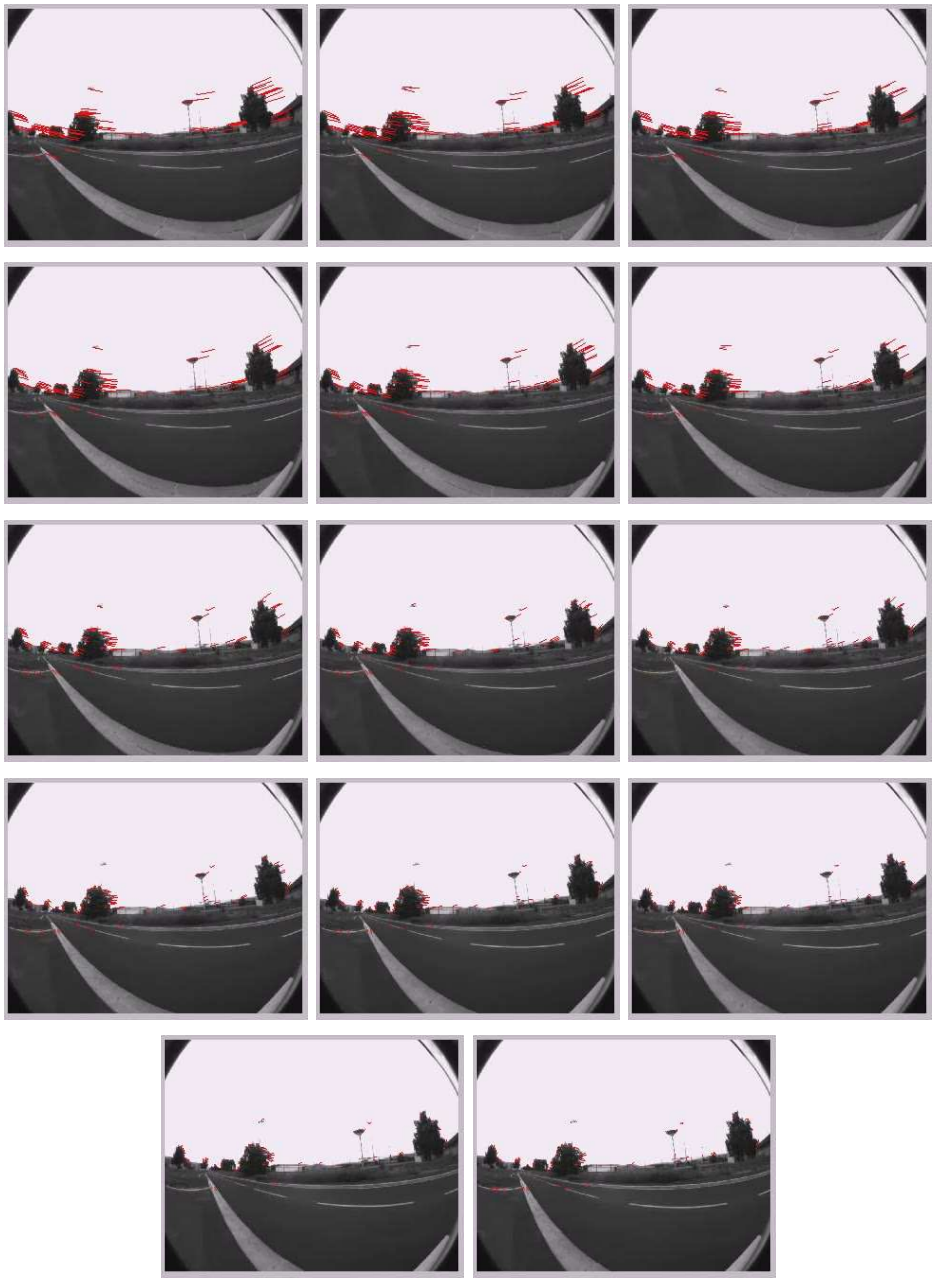


FIG. 5.19 – Séquence *Chemin simple* : images acquises successivement dans un virage et mises en correspondance avec une même image clé.

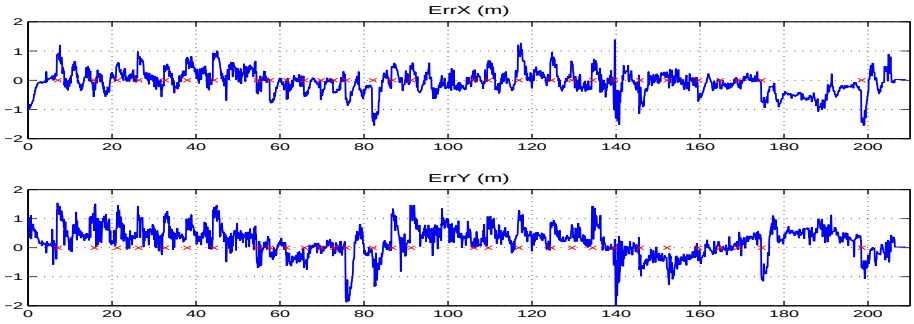


FIG. 5.20 – Séquence *Drone I* : erreurs de position $ErrX$ et $ErrY$ (exprimées en mètres) en fonction du temps (exprimé en secondes)

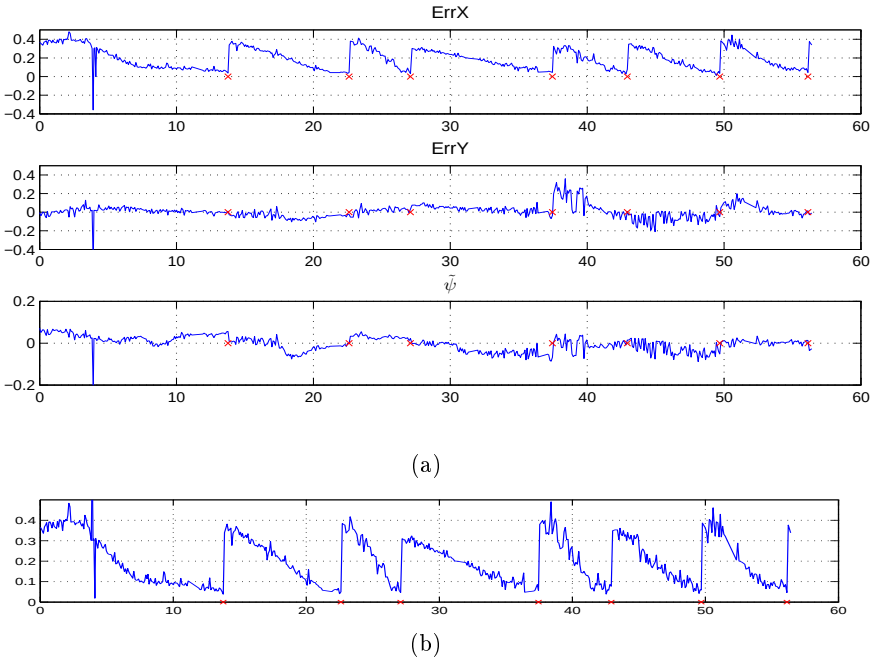
orientés : le drone a pu se déplacer d'images $\mathcal{I}_i$ à des images $\mathcal{I}_{i+1}$ mais également de $\mathcal{I}_{i+1}$ à $\mathcal{I}_i$ à plusieurs reprises lors de cette expérimentation.

5.4.3.2 Séquence *Drone II*

Nous considérons maintenant la séquence *Drone II*. Comme le sol n'est pas texturé, la Méthode 1 d'estimation de la vitesse de translation ne peut pas être employée (voir Section 5.2.2). Nous utilisons alors la Méthode 2. Le robot est téléopéré vers une position proche de la trajectoire apprise. Il est localisé sur l'image $\mathcal{I}_3$ (voir Fig. 5.13) puis un chemin contenant 8 images clés est extrait afin de rejoindre $\mathcal{I}_{10}$. Lors de la tâche de navigation, les erreurs de position ainsi que de lacet doivent être régulées à zéro.

En raison d'un problème technique, le mode automatique est arrêté, après que l'image $\mathcal{I}_9$ ait été atteinte. Les erreurs de position et de lacet sont représentées Fig. 5.21. On observe que l'erreur $ErrX$ décroît jusqu'à atteindre une valeur proche de 0.05 mètre lorsque les images clés sont atteintes et que cette erreur est toujours positive. Le drone ne dépasse donc pas les situations d'acquisition des images clés dans la direction X_c. Le chemin parcouru lors de l'apprentissage étant approximativement dirigé dans cette direction, de faibles erreurs de position le long de Y_c sont attendues. Cela est effectivement observé au début du chemin parcouru ($|ErrY| < 10$ centimètres) puis on observe des valeurs supérieures à 20 centimètres en valeur absolue à partir du temps t = 37 secondes. Ce comportement est du à un déplacement latéral du drone probablement causé par une perturbation extérieure. On observe, cependant, que cette erreur est régulée à zéro avant chacune des images clés suivantes. Quant à l'erreur de lacet, elle est régulée à zéro excepté à la première et à la quatrième image clé où l'erreur est inférieure à 0.1 radian en valeur absolue.

Les consignes du régulateur d'orientation sont représentées Fig. 5.22. On observe qu'elles sont souvent nulles. En effet, les consignes sont transmises au drone toutes les 60 millisecondes par le poste au sol alors que le temps moyen de calcul des entrées de commande par notre système de navigation est de 85 millisecondes dans cette expérimentation. Une valeur de consigne nulle est alors envoyée par défaut. Nous voyons ici que des algorithmes plus rapides seront à mettre en place pour éviter cette situation.



(a)

(b)

FIG. 5.21 – Séquence *Drone II*, exp. 1 : (a) erreurs de position $ErrX$ et $ErrY$ exprimées en mètre et erreur de lacet $\tilde{\psi}$ exprimée en radians, (b) $err = \| [ErrX\ ErrY]^T \|$ exprimée en mètres en fonction du temps (en secondes).

Au temps t = 3.9 secondes, le contenu de l'image acquise est très altéré (voir Fig. 5.23). Seuls 5 couples de points sont alors appareillés de façon robuste avec l'image désirée, ce qui conduit à une mauvaise estimation des entrées de commande comme on peut l'observer sur la Fig. 5.21 (a). Afin d'assurer la sécurité du drone, les commandes sont fixées à une valeur nulle lorsque ce cas de figure se présente (plus exactement, lorsque le nombre de points appareillés est inférieur à 10).

De manière générale, les résultats obtenus sur le drone sont satisfaisants au vu des conditions expérimentales difficiles. Tout d'abord, les hypothèses faites lors de la modélisation du drone (géométrie parfaite...) ne sont pas vérifiées en pratique. En outre, les retards imposés par les liaisons sont très importants. De plus, le fait d'utiliser des liaisons hertziennes pour rapatrier les images acquises par les caméras soit vers le PC (pour l'estimation des entrées de commande) soit vers le poste au sol (pour l'estimation de la vitesse de translation) engendre des retards, des images de mauvaises qualités (voir l'image utilisée pour estimer les entrées de commande représentée Fig. 5.23 et les images acquises pour estimer la vitesse de translation représentées Fig. 5.24 par exemple) voire même la perte d'images. Afin de résoudre ces problèmes, la solution idéale serait de réaliser tous les traitements en embarqué. Pour cela, les différents algorithmes devront être modifiés afin de réduire leur complexité et ainsi les rendre compatibles avec la faible puissance de calcul disponible sur le processeur embarqué du drone. Une

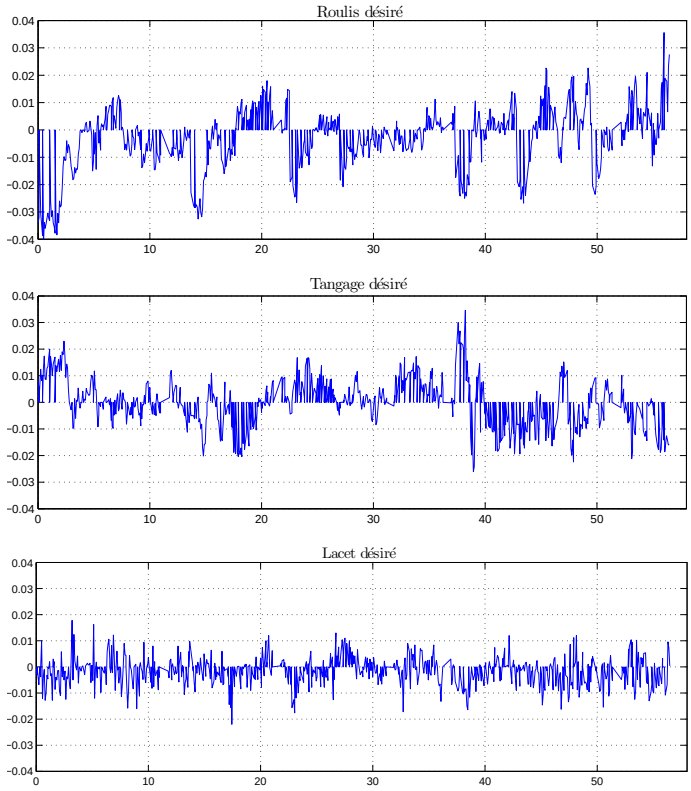


FIG. 5.22 – Séquence *Drone II*, exp. 1 : angles désirés (exprimés en radian) en fonction du temps (exprimé en secondes).

FIG. 5.23 – Séquence *Drone II*, exp. 1 : image au contenu très altéré, acquise au temps t= 3.9 secondes.

FIG. 5.24 – Séquence *Drone II*, exp. 1 : deux images de la caméra dirigée vers le sol.

autre approche consisterait à embarquer des processeurs plus performants. Toutefois, les solutions actuelles (type PC104 ou autre) ne sont, pour l'instant, guère compatibles avec la faible charge utile du drone.

5.5 Évaluation des performances

Après avoir valider notre stratégie de navigation dans différents environnements, les performances de notre stratégie sont ici évaluées à travers plusieurs indicateurs et expérimentations. Dans un premier temps, nous évaluons la gestion de la mémoire et les temps de calcul. Dans un second temps, nous donnons quelques éléments permettant de juger de la robustesse vis-à-vis des changements dans les images. Finalement, nous présentons des expérimentations dans des environnements de grande taille et nous évaluons la précision et la reproductibilité de notre stratégie à travers des expérimentations de navigation répétées et / ou en boucle.

5.5.1 Gestion de la mémoire

Le logiciel SOVIN a été conçu pour gérer la mémoire sensorielle correspondant à un environnement de grande taille. Nous présentons ici le contenu de la BDD intégrant l'ensemble des mémoires sensorielles apprises avec le RobuCab (se référer au Tableau 5.2). Elle contient environ 1.1 million d'objets qui représentent presque 1.5 Go de données. 68 séquences ont été sauvegardées et des images ont été acquises par 3 capteurs (une caméra, un DGPS et un odomètre) pour plus de 2 000 situations différentes. Avec ces capteurs ont été acquises 2 093 image visuelles (images en niveau de gris, au format PGM, de taille 800×600), 2 093 images contenant les coordonnées DGPS correspondant aux prises de vue de ces images et 1 324 images correspondant aux positions données par l'odométrie du véhicule pour certaines séquences. À raison de 500 points détectés par image, la mémoire contient plus de 1 million de points image. Les données les plus volumineuses sont les images et les points 2D (99% de la taille mémoire totale).

L'utilisation d'une BDD a permis de sauvegarder tous les éléments nécessaires à la navigation dans un environnement de grande taille. De plus, ceux-ci peuvent être

modifiés aisément via l'IHM. On peut, par exemple, supprimer des images, modifier des séquences, supprimer des capteurs, tout en préservant l'intégrité de la structure de la mémoire.

Dans un contexte réel d'application, il peut être nécessaire de limiter la taille de la mémoire. Pour cela, on pourrait tout d'abord envisager de ne pas sauvegarder le contenu global de l'image mais seulement une image en taille réduite permettant de visualiser son contenu. De plus, il apparaît que certains points image ne sont jamais utilisés pour la navigation (voir Annexe B). On pourrait alors envisager de les supprimer de la mémoire afin de limiter la taille mémoire. Cependant, les problématiques de perception sous-jacentes sont très complexes et doivent être étudiées spécifiquement.

Table	Nb éléments	Taille mémoire
JEUTEST	4	132 o
SEQUENCE	68	4.35 Ko
CONTIENT_SEQUENCE	68	630 o
A_POUR_SEQUENCE_SUIVANTE	75	684 o
ARC	2 025	26.02 Ko
CAPTEUR	3	180 o
NOEUD	2 093	18.62 Ko
CONTIENT_IMAGE_ACQUIS_PAR	5 510	70.28 Ko
IMAGE	5 510	964.69 Mo
POINT2D	1 046 500	512.56 Mo
POINT3D	89 456	7.67 Mo
Total	1 151 312	1485.05 Mo

TAB. 5.2 – État de la base de données contenant les mémoires sensorielles du RobuCab.

5.5.2 Temps de calcul

Afin de commander efficacement les plateformes robotiques utilisées, il est nécessaire que les entrées de commande soient estimées à une fréquence la plus élevée possible. Cela passe tout d'abord par des algorithmes de vision rapides. Nous utilisons une librairie développée par E. Royer au LASMEA dont le code de détection des points d'intérêt et d'appariement a été particulièrement optimisé en faisant usage de la parallélisation SIMD (*Single Instruction Multiple Data*). Les temps de détection et de mise en correspondance pour une image de taille 800×600 pixels sont les suivants :

Détection et description des points	35 ms
Mise en correspondance	10 ms

Le logiciel SOVIN a été développé afin de gérer de façon efficace les données de la mémoire sensorielle. Que ce soit lors de la localisation initiale, lors de l'estimation des entrées de commande ou pour la visualisation de ces données, il est nécessaire que

la récupération soit rapide. Les temps moyens de récupération, pour la BDD décrite précédemment, sont les suivants :

Récupération d'une image de taille 800×600 pixels décrite en niveau de gris	4 ms
Récupération du descripteur global	1-2 ms
Récupération des 500 points image à partir de l'identifiant de l'image	25 ms

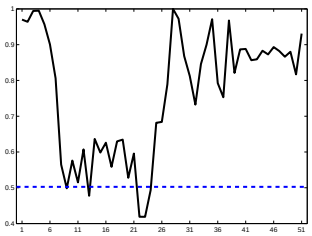
On observe que le temps de récupération des points d'une image est plus rapide que la détection et la description de ceux-ci à partir de l'image. Il est donc préférable de sauvegarder les points dans la BDD plutôt que ne sauvegarder que les images puis de détecter les points ensuite. Cela devrait être encore plus avantageux si la détection et la description des points sont réalisées avec une méthode avec un coût calculatoire élevé. À travers ces résultats, on remarque que l'étape de localisation initiale peut être réalisée très rapidement. Nous vérifions cela avec un exemple. Le RobuCab se situe initialement près du lieu Tramway et acquiert l'image $\mathcal{I}_c$ représentée Fig. 5.25. La séquence sur laquelle le véhicule se situe est désignée manuellement. Le robot est ensuite localisé parmi les 51 images de cette séquence. Les temps et résultats sont les suivants :

	Temps moyen par image	Résultat
Approche globale	2 ms	$\mathcal{I}_{23}$
Approche locale	33 ms	$\mathcal{I}_{10}$
Approche hiérarchique	7 ms	$\mathcal{I}_9$

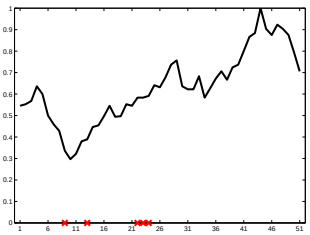
Avec l'approche hiérarchique proposée Section 4.2.3, la localisation est obtenue en 360 ms (5 images clés ont été sélectionnées à partir des descripteurs globaux avec le seuil τ=1.2). Le résultat ($\mathcal{I}_9$) est relativement similaire à l'image courante $\mathcal{I}_c$ (voir Fig. 5.25). Si l'approche globale est employée seule, la localisation est obtenue en 103 ms mais le résultat obtenu ($\mathcal{I}_{23}$) est très différent de l'image courante. Avec une approche locale seule, la localisation est obtenue en 1 680 ms et le résultat est très similaire à $\mathcal{I}_c$. Nous observons des temps de calcul similaires en comparant l'image courante avec un ensemble de 1 390 images de la BDD, excepté pour l'approche hiérarchique qui est plus rapide (car peu d'images ont été sélectionnées pour la seconde phase de la stratégie).

5.5.3 Robustesse

Nous avons pu mener un grand nombre d'expérimentations dans des lieux variés avec différents types de robots. C'est la répétition de telles expérimentations qui permet de tirer des conclusions quant à la robustesse du système de navigation proposé. Par exemple, par le suivi de la séquence *Grand Environnement* lors de la première expérimentation (qui sera détaillée dans la Section 5.5.4), nous pouvons nous faire une idée de la robustesse vis-à-vis des modifications dans l'image entre les phases d'apprentissage et de navigation autonome. La Figure 5.26 montre plusieurs couples d'images ($\mathcal{I}_k^*$, $\mathcal{I}_k$) où $\mathcal{I}_k^*$ dénote une image clé et $\mathcal{I}_k$ l'image correspondante acquise lors de la phase autonome. On

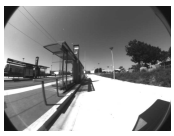


(a) Distance entre les descripteurs globaux



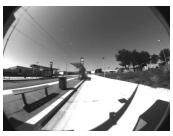
(b) Distance entre les descripteurs locaux

(c) $\mathcal{I}_c$



(d) $\mathcal{I}_9$ (loc. hiér.)

(e) $\mathcal{I}_{10}$ (loc. locale.)



(f) $\mathcal{I}_{13}$

(g) $\mathcal{I}_{22}$

(h) $\mathcal{I}_{23}$ (loc. globale)

(i) $\mathcal{I}_{24}$

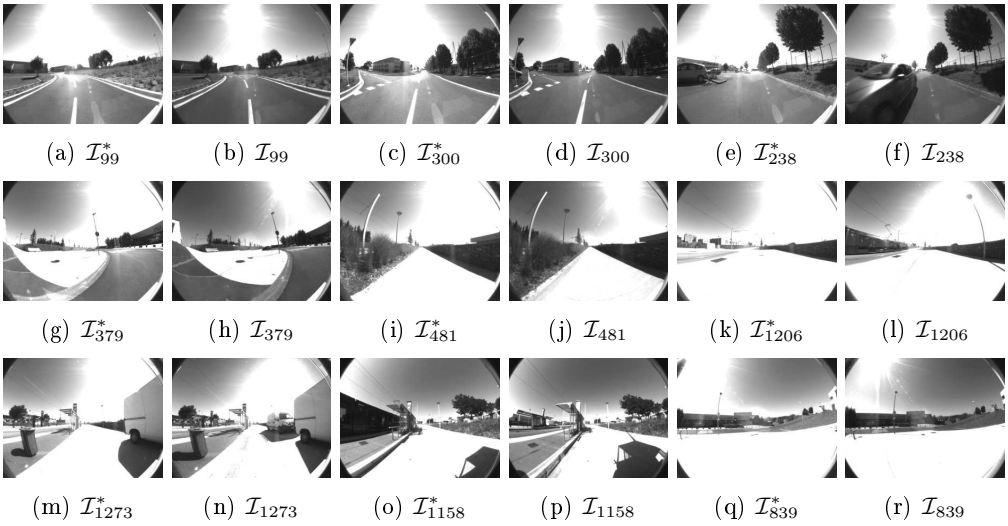
FIG. 5.25 – Étape de localisation. L'image la plus proche de $\mathcal{I}_c$ est $\mathcal{I}_{23}$ avec l'approche globale, $\mathcal{I}_{10}$ avec l'approche locale et $\mathcal{I}_9$ avec l'approche hiérarchique (les images $\mathcal{I}_9$, $\mathcal{I}_{13}$, $\mathcal{I}_{22}$, $\mathcal{I}_{23}$ et $\mathcal{I}_{24}$ ont été sélectionnées avec la méthode globale).

peut observer de nombreuses modifications dans les images. Pour les couples $(\mathcal{I}^*_{99}, \mathcal{I}_{99})$, $(\mathcal{I}^*_{300}, \mathcal{I}_{300})$, $(\mathcal{I}^*_{379}, \mathcal{I}_{379})$, $(\mathcal{I}^*_{481}, \mathcal{I}_{481})$ et $(\mathcal{I}^*_{839}, \mathcal{I}_{839})$, les conditions d'éclairage sont modifiées. Pour les couples $(\mathcal{I}^*_{238}, \mathcal{I}_{238})$, $(\mathcal{I}^*_{1273}, \mathcal{I}_{1273})$, $(\mathcal{I}^*_{1206}, \mathcal{I}_{1206})$ et $(\mathcal{I}^*_{1158}, \mathcal{I}_{1158})$, on observe que des véhicules sont apparus tandis que d'autres ont disparus. On observe également que pour les couples $(\mathcal{I}^*_{1206}, \mathcal{I}_{1206})$ et $(\mathcal{I}^*_{839}, \mathcal{I}_{839})$, les images contiennent très peu de zones texturées (condition nécessaire pour détecter les points d'intérêt). Malgré ces changements d'illumination et de contenu, la tâche de navigation a pu être réalisée correctement.

De la même manière, lors du suivi de la séquence *Drone I* par le drone (se référer à la Section 5.4.3.1), on observe de nombreux changements dans les images entre les phases d'apprentissage et de suivi. Par exemple, sur les images représentées Fig. 5.27, un véhicule a disparu. On observe également que l'image courante représentée Fig. 5.27 (a) est de mauvaise qualité. Malgré cela, le chemin sensoriel a pu être suivi presque entièrement.

5.5.4 Navigation dans des environnements de grande taille

Comme nous l'avons dit dans le paragraphe 5.5.2, les temps de calcul et de récupération des données sont compatibles avec une expérimentation le long d'un chemin visuel de taille importante. Nous proposons maintenant des expérimentations sur le RobuCab afin de valider notre stratégie complète de navigation dans ce contexte.

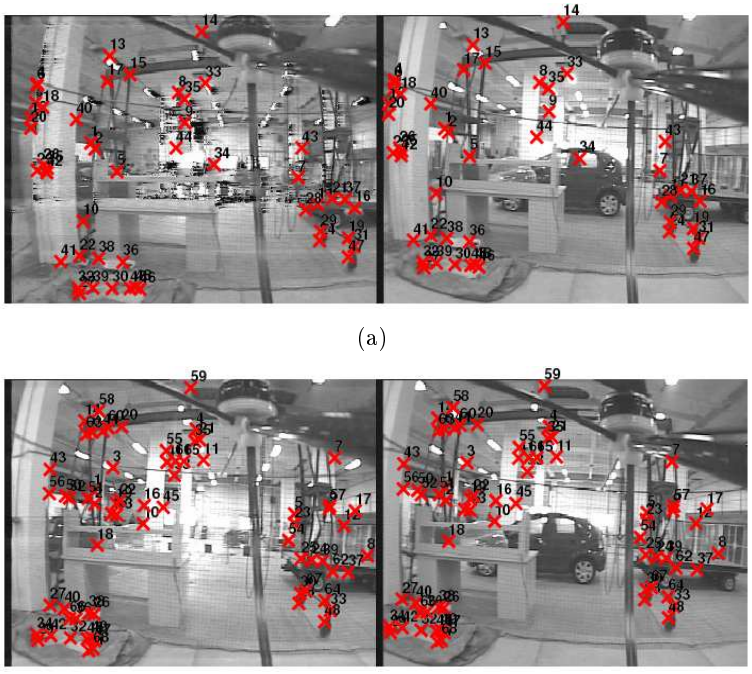


(a) $\mathcal{I}^*_{99}$ (b) $\mathcal{I}_{99}$ (c) $\mathcal{I}^*_{300}$ (d) $\mathcal{I}_{300}$ (e) $\mathcal{I}^*_{238}$ (f) $\mathcal{I}_{238}$

(g) $\mathcal{I}^*_{379}$ (h) $\mathcal{I}_{379}$ (i) $\mathcal{I}^*_{481}$ (j) $\mathcal{I}_{481}$ (k) $\mathcal{I}^*_{1206}$ (l) $\mathcal{I}_{1206}$

(m) $\mathcal{I}^*_{1273}$ (n) $\mathcal{I}_{1273}$ (o) $\mathcal{I}^*_{1158}$ (p) $\mathcal{I}_{1158}$ (q) $\mathcal{I}^*_{839}$ (r) $\mathcal{I}_{839}$

FIG. 5.26 – Robustesse face aux changements dans l'environnement : l'image $\mathcal{I}_k$ est l'image acquise lors de la phase autonome lorsque l'image $\mathcal{I}^*_k$ de la mémoire est atteinte.



(a)

(b)

FIG. 5.27 – Appariements robustes entre l'image courante (à gauche) et l'image clé à rejoindre qui est (a) l'image $\mathcal{I}_5$ et (b) l'image $\mathcal{I}_6$.

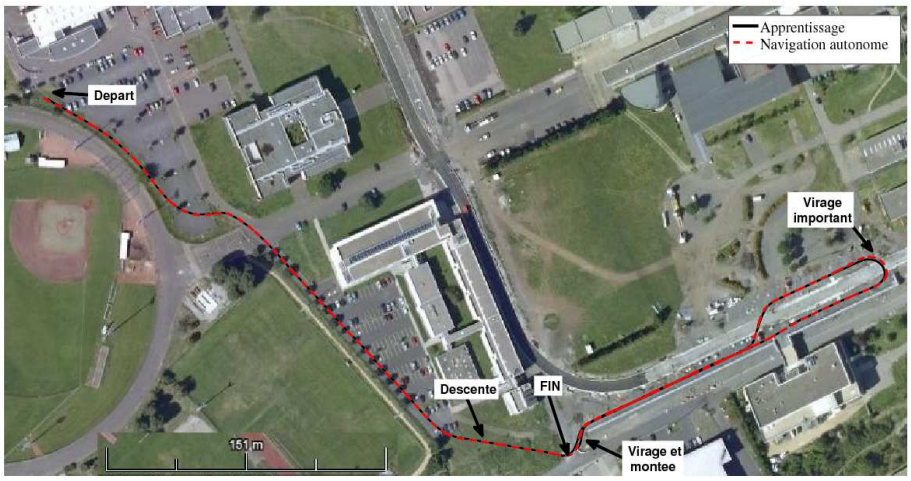


FIG. 5.28 – Mémoire *Grand Environnement*, exp. 1 : chemins parcourus lors de l'apprentissage et lors du suivi (représentés en coordonnées métriques).

Le RobuCab est amené initialement près de la séquence Γ^1 de la mémoire sensorielle *Grand Environnement* (voir Fig. 5.9). L'objectif est de réaliser entièrement la grande boucle (1 200 mètres de long) puis de suivre à nouveau cette boucle jusqu'au lieu *Tramway* soit un trajet de plus de 1 700 mètres. Le véhicule est tout d'abord localisé en comparant l'image initiale avec les 15 images de la séquence Γ^1. Le chemin visuel de référence contient 38 séquences (soit plus de 1 000 images clés). Deux expérimentations sont présentées ici. Lors de la première expérimentation, la tâche de navigation échoue. Le robot est stoppé au lieu *IFMA* après un trajet de 740 mètres parcouru en 13 minutes (soit une vitesse moyenne de 0.8 m/s) en raison d'un nombre insuffisant de primitives appareillées avec une des images clés. Le trajet parcouru de façon autonome est représenté Fig. 5.28[6]. Lors de cette expérimentation, 123 couples de points en moyenne ont été appareillés de façon robuste, avec un temps de calcul moyen de 82 ms / image. L'écart latéral par rapport au chemin de référence, estimé avec le DGPS, a pour moyenne 25 centimètres et écart type 35 centimètres. La médiane n'est que de 10 centimètres. Le zoom représenté Fig. 5.29 permet d'apprécier le comportement du véhicule lors du virage en U *Tramway*. On observe que le véhicule revient sur le chemin appris, après s'en être écarté au plus fort du virage.

La seconde expérimentation est réalisée quelques jours plus tard avec la même mémoire visuelle et le même objectif de navigation. Le robot est positionné au début de la séquence Γ^1. Lors de cette seconde expérimentation, le véhicule suit entièrement le chemin sensoriel (voir Fig. 5.31). L'expérimentation a duré 26 minutes pour une distance totale parcourue de 1 700 mètres. Lors de cette expérimentation, 115 couples de points en moyenne ont été appareillés de façon robuste, avec un temps de calcul moyen de 79

[6] Pour plus de clarté, seule la partie du chemin appris correspondant au chemin réalisé a été représentée.

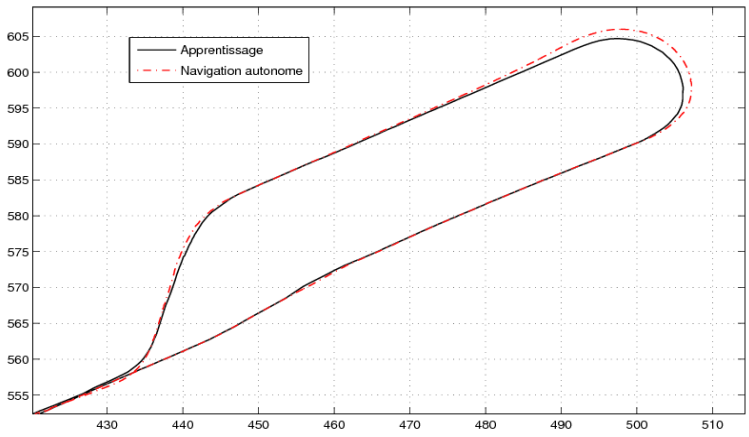


FIG. 5.29 – Mémoire *Grand Environnement*, exp. 1 : zoom sur les chemins vers le virage en U *Tramway* (représentés en coordonnées métriques)

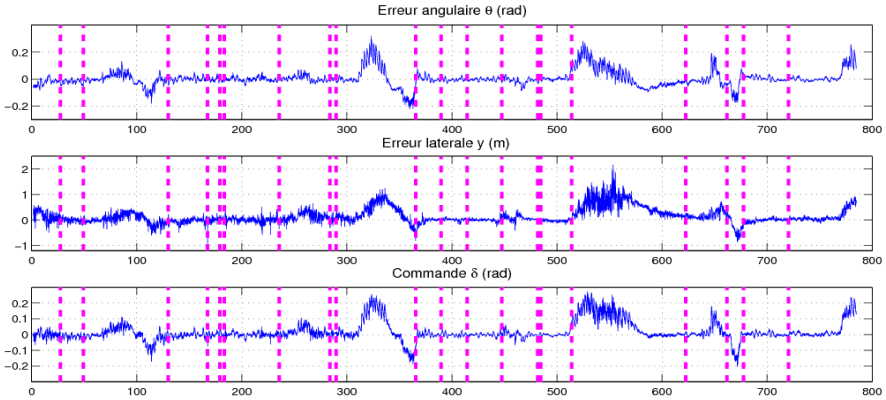
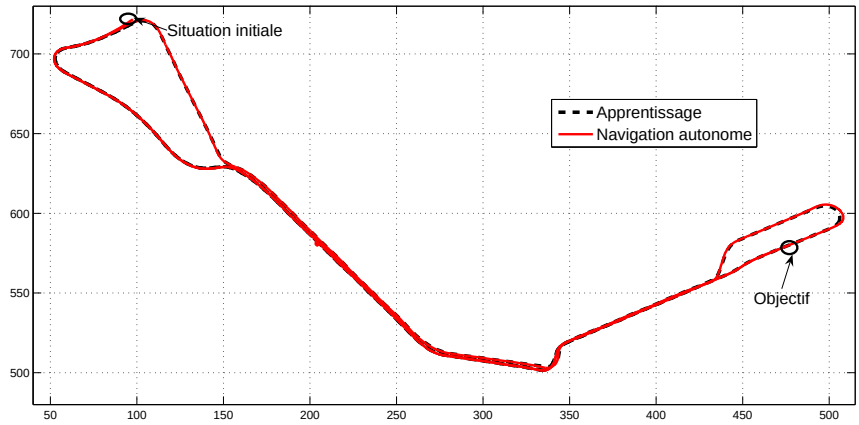
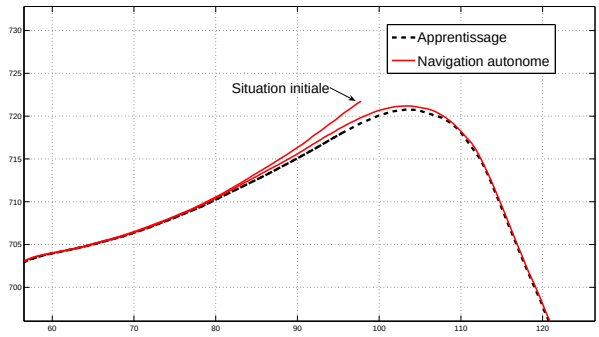


FIG. 5.30 – Mémoire *Grand Environnement*, exp. 1 : erreur angulaire (exprimée en radians), erreur latérale (exprimée en mètres) et commande (exprimée en radians) en fonction du temps (exprimé en secondes).



(a)



(b)

FIG. 5.31 – Mémoire *Grand Environnement*, exp. 2 : (a) chemins parcourus lors de l'apprentissage et lors de la phase de navigation autonome (représentés en coordonnées métriques) et (b) zoom sur les trajectoires au niveau de la position initiale.

ms / image. L'écart entre le chemin appris et le chemin désiré est de 22 centimètres en moyenne avec un écart type de 30 centimètres. Les erreurs sont représentées Fig. 5.32. Elles décroissent jusqu'à zéro excepté dans les virages importants.

Nous avons ici montré que notre approche peut être employée dans des environnements de grande taille. De plus, on note que la navigation autonome a pu être réalisée quelques jours après la première expérimentation (et après la phase d'apprentissage), sans modification de la mémoire. Des expérimentations plus poussées doivent être menées afin d'étudier la robustesse de notre stratégie au cours du temps.

À notre connaissance, ce trajet de 1 700 mètres parcouru de manière autonome en utilisant uniquement une caméra est le plus important reporté dans la littérature.

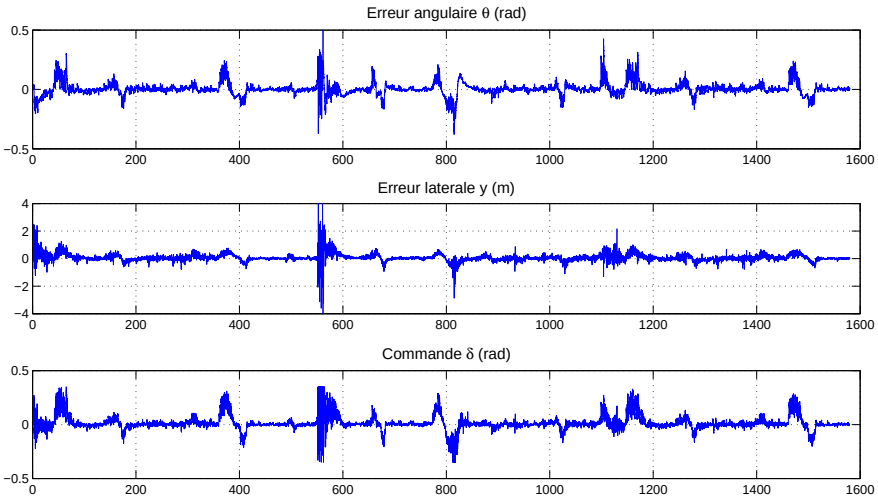


FIG. 5.32 – Mémoire *Grand Environnement*, exp. 2 : erreur angulaire (exprimée en radians), erreur latérale (exprimée en mètres) et commande (exprimée en radians) en fonction du temps (exprimé en secondes)..

5.5.5 Bouclage

Le cas d'une séquence en boucle est intéressant car c'est un bon moyen de visualiser les performances des stratégies de navigation. De manière remarquable, la carte topologique de la mémoire sensorielle définit explicitement ce bouclage. C'est un avantage certain de notre approche par rapport aux méthodes basées sur une représentation métrique de l'environnement, sujette à des dérives potentielles importantes. L'expérimentation qui suit est réalisée le long du chemin *Boucle CUST* (voir Section 5.3.2). Le chemin à suivre est défini à partir de la concaténation des séquences Γ^1, Γ^2 et Γ^3. Le point à vérifier est que le robot rejoint la situation correspondant à la première image de Γ^1 à la fin d'une "boucle". Nous vérifions ce point en donnant pour objectif au RobuCab de suivre cinq fois de suite le chemin *Boucle CUST*.

Les chemins parcourus sont représentés Fig. 5.33. On observe que, pour chaque boucle, la tâche de suivi est réalisée entièrement. Le véhicule se situe alors à la position correspondant à la première image de *Boucle CUST* et il peut alors suivre à nouveau le chemin sensoriel.

5.5.6 Reproductibilité

Un autre point intéressant pour visualiser les performances de notre stratégie est de vérifier que, pour un même chemin sensoriel, les trajectoires parcourues lors de la phase de navigation automatique sont similaires (reproductibilité). Revenons sur l'expérimentation précédente. Comme nous l'avons vu en simulation et lors de l'expérimentation le long de la séquence *Chemin Simple*, le long d'une ligne droite, l'écart entre les tra-

	$\mathcal{I}_3$ à $\mathcal{I}_4$	$\mathcal{I}_4$ à $\mathcal{I}_5$	$\mathcal{I}_5$ à $\mathcal{I}_6$	$\mathcal{I}_6$ à $\mathcal{I}_7$	$\mathcal{I}_7$ à $\mathcal{I}_8$	$\mathcal{I}_8$ à $\mathcal{I}_9$	$\mathcal{I}_9$ à $\mathcal{I}_{10}$
Exp. 1	8.8	4.5	10.4	5.5	6.7	6.4	-
Exp. 2	15.3	3.7	8.0	6.1	7.7	4.7	7.6
Exp. 3	10.2	3.1	11.0	4.5	5.8	4.5	4.4

TAB. 5.3 – Séquence *Drone II* : temps (exprimé en secondes) mis pour parcourir le chemin entre deux images clés pour les 3 expérimentations.

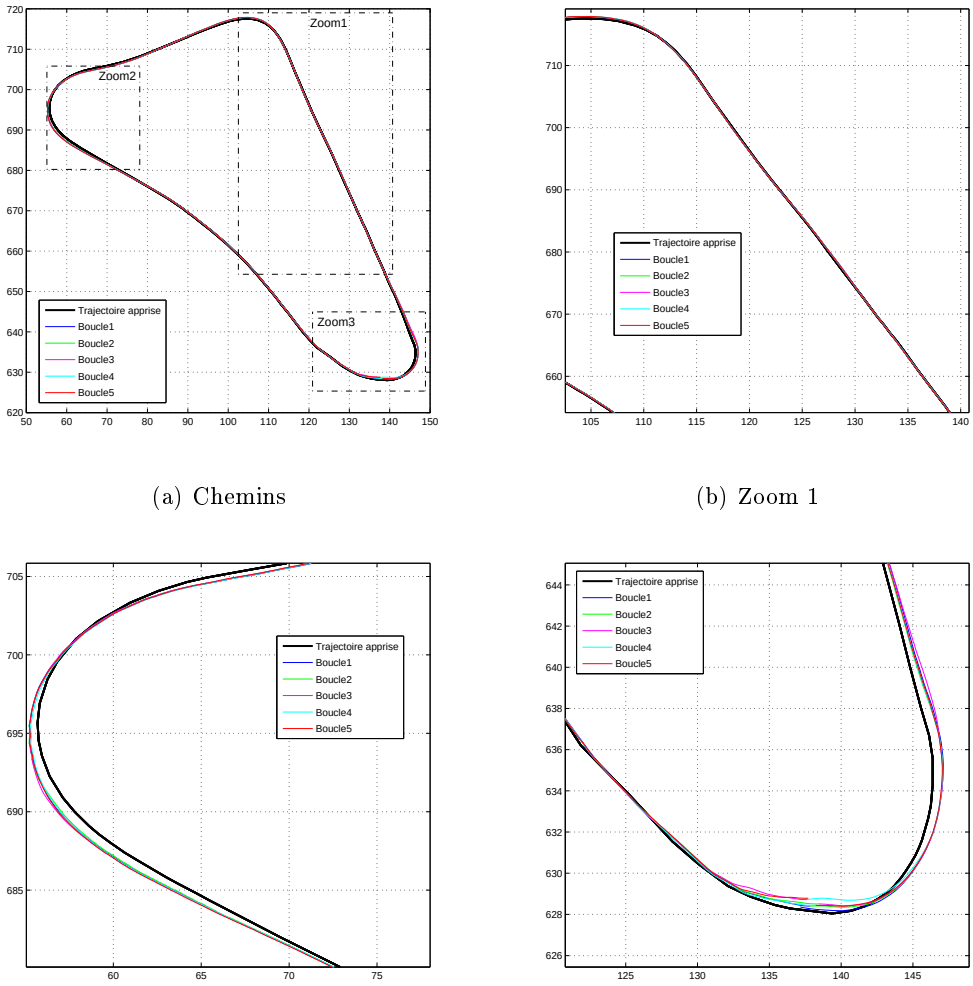
jectoires parcourues lors des phases de navigation autonome et d'apprentissage est nul (voir Zoom 1 Fig. 5.33) tandis que dans les virages serrés, cet écart est important (il atteint ici 70 centimètres, voir Zoom 2 Fig. 5.33).

Par contre, on observe que l'écart entre les chemins parcourus de façon autonome est faible (voir les zooms Fig. 5.33). Cela est vrai excepté sur la partie du chemin représentée Zoom 3 (Fig. 5.33) correspondant au lieu de changement de boucle. On observe, cependant, que le véhicule revient sur la même trajectoire ensuite. Comme les trajectoires parcourues sont similaires, les erreurs latérale et angulaire ainsi que la commande, représentées Fig. 5.34, sont similaires pour chaque boucle (sur cette figure, les segments verticaux marquent le début d'une nouvelle boucle).

Nous étudions maintenant la reproductibilité du suivi de chemin dans un autre contexte en répétant deux nouvelles fois l'expérimentation menée sur le drone le long de la séquence *Drone II*. Le robot est téléopéré vers une position initiale approximativement identique à celle de la première expérimentation. Dans les deux cas, la tâche de navigation est entièrement réalisée (l'image $\mathcal{I}_{10}$ est atteinte). Les évolutions de l'erreur ($err = \| [ErrX\ ErrY]^T \|$) sont très similaires lors des trois expérimentations (voir Figure 5.35). Selon les conditions expérimentales, le temps mis pour parcourir le chemin entre deux images clés successives est différent (voir Tab. 5.3). En effet, ce temps dépend entre autres des conditions expérimentales (amplitude des perturbations, qualité des images, état de charge de la batterie...).

5.6 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté la mise en œuvre de notre système de navigation. Celle-ci a largement été simplifiée par l'utilisation d'un logiciel (SOVIN) développé dans le cadre de nos travaux. Nous avons ensuite présenté plusieurs expérimentations permettant de valider notre stratégie complète de navigation. Nous avons tout d'abord montré que la tâche de navigation pouvait être réalisée avec différentes caméras, dans différents types d'environnement et avec les trois plateformes expérimentales considérées. Nous avons ensuite vu que notre stratégie peut être employée à l'échelle d'un campus tout en respectant les fermetures de boucle. Nous avons également montré la robustesse de notre approche vis-à-vis de changements dans l'im-



(a) Chemins (b) Zoom 1

(c) Zoom 2 (d) Zoom 3

FIG. 5.33 – *Boucle CUST* : chemins (exprimés en coordonnées métriques) parcourus lors de la phase d'apprentissage et lors des phases de suivi autonome.

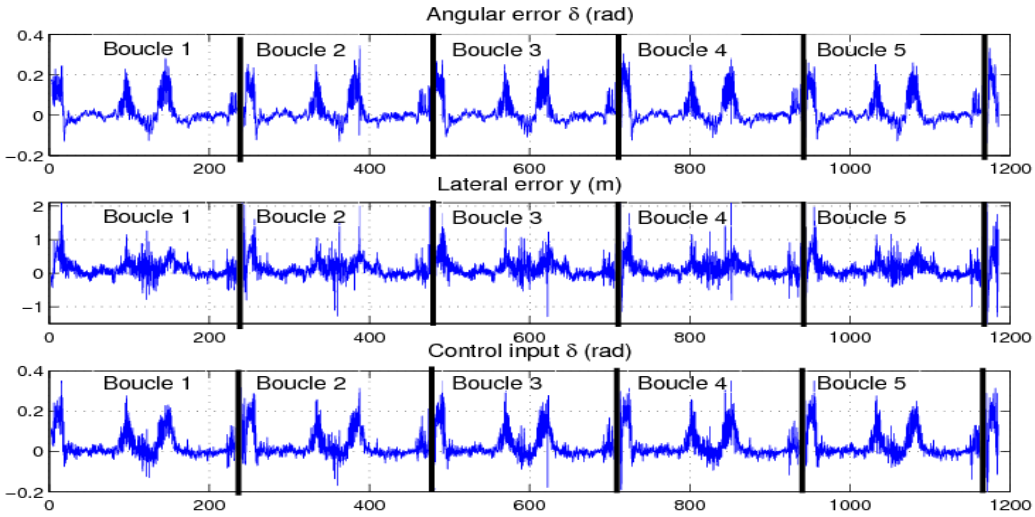
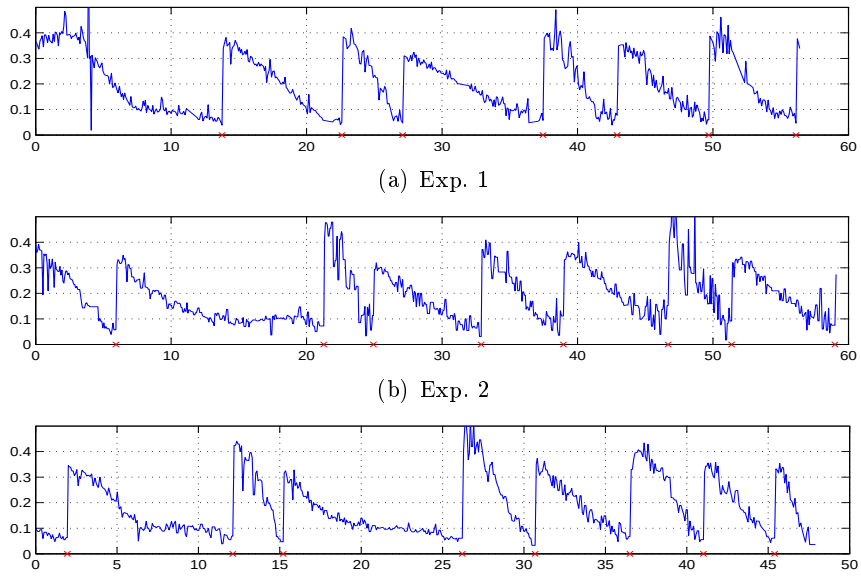


FIG. 5.34 – *Boucle CUST* : erreur angulaire θ (exprimée en radians), erreur latérale y (exprimée en mètres) et commande (exprimée en deg/s) en fonction du temps (en secondes).



(a) Exp. 1

(b) Exp. 2

(c) Exp. 3

FIG. 5.35 – Séquence *Drone II* : valeur du critère de changement d'image clé $err = \| [ErrX\ ErrY]^T \|$ exprimée en mètres en fonction du temps (en secondes) pour les trois expérimentations.

age.

Difficultés rencontrées

Comme nous l'avons vu lors de la description des différentes expérimentations, certaines situations rendent difficile la réalisation de la tâche de navigation. Cela est généralement du à la combinaison de plusieurs facteurs (souvent très couplés). On peut tout de même distinguer deux catégories : les difficultés liées aux stratégies de perception et celles liées aux stratégies de commande.

Problèmes liés à la perception La première difficulté est liée à la qualité des images qui peut être détériorée. Cette situation est notamment rencontrée lorsque les images sont transmises par liaison HF comme dans nos expérimentations sur le drone. Un premier remède consiste à utiliser un transmetteur plus performant. Cependant, le choix d'un tel transmetteur est contraint par la masse utile embarquable et par les ressources énergétiques disponibles sur le drone. Dans le même ordre d'idée, il est également possible de coupler un amplificateur de signal au transmetteur. Cette dernière solution a été testée sans apporter totale satisfaction. Une alternative consiste à traiter les images en embarqué. Cette solution soulève une nouvelle fois la question des ressources disponibles pour effectuer les traitements.

La seconde source de difficulté rencontrée provient de la mauvaise adéquation des opérateurs de traitement d'image bas niveaux aux capteurs utilisés. Par exemple, pour l'expérimentation réalisée avec une caméra catadioptrique (décrite Section 5.4.1), les lieux d'acquisition des images clés sont situés tous les 15 centimètres en moyenne car le voisinage des points dans l'image varie très rapidement pour des points 3D proches de la caméra. Cette distance inter-image relativement faible conduit au stockage d'informations inutiles. De plus, lorsque la Stratégie 1 est employée, elle conduit également à un changement fréquent de consigne entraînant des discontinuités fréquentes de la commande. La solution à ce problème réside dans la définition d'opérateurs bas niveaux parfaitement adaptés aux images panoramiques et utilisable dans des applications temps réel. Cette problématique est encore largement ouverte.

Une troisième difficulté est liée aux variations de luminosité dans la scène. L'algorithmique actuelle est robuste dans une certaine limite à celles-ci. Toutefois, l'algorithme d'appariement peut être mis en difficulté lorsque ces variations ne sont pas homogènes dans le voisinage des points considérés. Ce problème peut être traité soit par l'utilisation d'algorithmes efficaces de ce point de vue ou soit par l'utilisation de caméras plus performantes. En particulier, des problèmes de saturation dans l'image peuvent être causés par la faible dynamique des caméras actuelles. En pratique, il s'agit de trouver un couple {ouverture du diaphragme, temps d'exposition} qui permet d'offrir un bon compromis à la fois pour une même scène contenant des zones très illuminées et des zones à l'ombre mais également pour toute la séquence d'images. Ce problème peut être résolu en partie en choisissant une caméra ayant une dynamique plus importante ou en utilisant une caméra *intelligente* dont les caractéristiques changent automatiquement en fonction de la scène (voir, par exemple, [Brame 96]).

Enfin, une quatrième source de difficulté rencontrée provient des modifications qui interviennent dans la scène entre la phase d'apprentissage et la phase de navigation autonome. À travers les différentes expérimentations menées, nous avons vu que l'approche était relativement robuste vis-à-vis de ce genre de situation. Une mémoire visuelle peut alors être employée durant un certain temps. Afin d'augmenter la robustesse de l'approche vis-à-vis des changements de contenu, deux solutions sont dors et déjà envisageables. La première solution possible consiste à mettre à jour la mémoire sensorielle lors des phases de navigation autonome. De nouvelles images pourraient être ajoutées à la mémoire sensorielle et d'autres supprimées en fonction des observations capteurs courantes. La seconde solution consiste à séparer la scène en trois parties : les éléments fixes à long terme (comme les bâtiments, le bord des routes, du couloir), les éléments fixes à court terme (végétation) et la partie mobile (piétons, personne dans le champ de vue, autres robots). Les éléments fixes sont alors les éléments de référence à prendre en compte pour estimer correctement les variables d'état de la commande tandis que les éléments fixes à court terme (supposés statiques lors d'une phase de navigation) permettraient d'obtenir le mouvement propre de la caméra.

Problèmes liés à la commande Pour les robots à roues, les résultats sont relativement satisfaisants malgré la présence de nombreuses perturbations (glissements, erreurs d'estimation des entrées de commandes...). Ces performances sont, cependant, à nuancer, surtout dans le cas du RobuCab. En effet, de par son inertie, celui-ci filtre assez largement ces perturbations. Dans le cas du drone, nous avons réalisé des expérimentations dans un environnement d'intérieur fermé où il n'y a théoriquement pas de vent. En pratique, nous avons observé la présence de nombreuses perturbations (vent créé par la rotation des hélices du drone, effet de sol) qui empêchent alors le bon déroulement de la tâche de navigation autonome. Afin d'envisager des expérimentations en extérieur avec ce type d'engin, il est donc nécessaire de construire des schémas de commande robustes vis-à-vis de ces perturbations.

Une autre difficulté importante est l'estimation de la vitesse de translation du drone. Les deux méthodes proposées utilisent une caméra dirigée vers le sol supposé plan. La Méthode 1 fonctionne convenablement dans ces conditions mais nécessite l'observation d'un sol texturé. La Méthode 2 est plus difficile à mettre en œuvre et ne fonctionne que lorsque le sol est quasiment plan. Dans les deux cas, l'utilisation d'une caméra supplémentaire est contraignante car elle surcharge l'engin. Il serait souhaitable d'approfondir notre étude sur ce point afin d'aboutir à un estimateur précis de vitesse employant la caméra déjà utilisée pour la navigation. Une autre piste de développement est la construction d'un schéma de commande ne nécessitant pas l'estimation de la vitesse de translation.

Chapitre 6

Conclusion et perspectives

Le travail que nous avons présenté dans ce mémoire avait comme première ambition de rendre les robots mobiles plus autonomes. Cette ambition nous a amenés à proposer une stratégie globale de navigation basée sur la représentation de l'environnement par une mémoire sensorielle.

Bilan

Notre contribution principale est la conception d'une stratégie complète de navigation pour les robots mobiles allant de l'apprentissage de site à la commande automatique. De manière très synthétique, la stratégie adoptée se divise en trois étapes.

Dans la première étape, la mémoire sensorielle du robot est constituée. Pour cela, le robot se déplace dans son environnement de travail et acquiert des images représentant cet environnement tel qu'il est perçu par le capteur embarqué. Les données sont triées et structurées dans une base de données : la mémoire sensorielle. Cette représentation permet au robot d'alimenter tous les processus nécessaires à sa navigation. De manière assez remarquable, elle intègre implicitement les notions de navigabilité et de commandabilité et donne également une vue globale de l'environnement aisément exploitable pour la planification de chemins vers des objectifs lointains. En outre, elle peut également être vue comme un moyen de communication entre l'utilisateur et le robot

La deuxième étape consiste à déterminer la localisation initiale du robot en comparant l'image courante acquise par le capteur aux images de la mémoire. Nous avons proposé une stratégie de localisation hiérarchique lorsque le capteur est une caméra grand-angle. Celle-ci se base sur une mise en correspondance de descripteurs globaux afin de réaliser une première sélection des images clés potentielles. Dans un second temps, le résultat de la localisation est obtenu via l'appariement de points d'intérêt entre l'image courante et les images sélectionnées. Cette approche semble être le meilleur compromis entre précision, quantité de données à mémoriser et coût calculatoire dans notre étude comparative.

Dans la dernière étape, le chemin sensoriel permettant d'aller de la position courante à un objectif défini dans la mémoire, est extrait. Ce chemin, constitué d'un ensemble

de données capteurs, est ensuite suivi par le robot via un schéma de commande basé sur les concepts de la commande référencée capteurs. Dans ce contexte, des stratégies de commande ont été développées pour les robots mobiles de type char et bicyclette et pour les drones de type quadrirotor. Des simulations ont permis de les évaluer et de souligner leurs limites. Ces lois de commande sont alimentées par des variables d'état calculées à partir de la mise en correspondance de l'observation courante avec les données successives du chemin sensoriel. Afin de réaliser l'estimation des variables d'état, sans aucune modification algorithmique, pour une classe importante de capteurs visuels (caméras conventionnelles, caméras catadioptriques et caméras à optique fisheye), nous avons montré l'adéquation (théorique et pratique) du modèle de projection unifié, originellement développé pour les caméras catadioptriques centrales, pour les caméras fisheye. Ce résultat est très intéressant car il permet d'étendre de manière immédiate les techniques de reconstruction euclidienne partielle, exploitées pour l'estimation du vecteur d'état des systèmes robotiques considérés, au cas d'une observation de la scène par une caméra fisheye.

D'un point de vue expérimental, notre objectif était de réaliser des tâches de navigation dans des environnements de taille importante (par exemple, à l'échelle d'un campus ou d'une ville). Pour cela, nous avons développé le logiciel nommé SOVIN (pour *SOftware for VIsual Navigation*). À l'aide de ce logiciel, la stratégie de navigation par mémoire sensorielle a pu être validée pour différents types de robots mobiles à roues en milieu intérieur et extérieur et pour un robot volant de type quadrirotor en milieu intérieur. Nos expérimentations ont également été conduites avec différents types de caméras (caméras catadioptriques et fisheye). En particulier, les dernières démonstrations sur le campus des Cézeaux ont montré l'efficacité de l'approche proposée pour des tâches de navigation en boucle et sur des distances importantes.

Perspectives

Comme nous l'avons évoqué dans le Chapitre 2, certaines étapes de notre stratégie de navigation sont réalisées manuellement, notamment lors de la phase d'apprentissage. Afin d'accroître encore l'autonomie du système robotique, deux principaux points sont à traiter en priorité.

Le premier point concerne le remplacement de l'étape de téléopération par une exploration automatique de l'environnement. Cette thématique est vaste et soulève de nombreuses problématiques. En effet, le système robotique devra alors être capable de se déplacer de façon sûre dans un environnement inconnu. De plus, il serait souhaitable que tout l'espace de navigation du robot soit complètement exploré. En outre, les chemins parcourus lors de cette phase d'exploration doivent être exploitables lors du suivi automatique, le comportement obtenu lors de la navigation autonome étant fortement lié à celui du robot lors de l'apprentissage. Pour répondre à ces besoins, une solution envisageable consiste à déplacer le robot le long des branches du diagramme de Voronoï comme il a été proposé dans [Victorino 02]. La stratégie développée dans ces travaux

est intéressante pour plusieurs raisons. Tout d'abord, elle ne nécessite pas de définir explicitement ce diagramme dans une carte métrique. De plus, l'utilisation d'un schéma de commande basé sur le concept de la commande référencée capteurs permet un suivi robuste vis-à-vis des erreurs de modélisation et des bruits de mesure. Dans ce contexte, une autre piste pour l'exploration de l'environnement est l'utilisation de gabarits de lieux et de schémas généraux comme cela est fait par les êtres humains. Il s'agit alors de développer des outils pour construire et exploiter ces gabarits et schémas.

Le second point à traiter pour automatiser la phase d'apprentissage est la mise à jour des cartes topologiques sans intervention humaine. À cette fin, il serait judicieux d'inclure à notre système de navigation des outils de détection robuste des lieux de l'environnement déjà mémorisés, en s'inspirant par exemple de la stratégie de détection visuelle de fermeture de boucle mise en œuvre dans [Angeli 08a].

Une autre perspective intéressante est l'enrichissement de la mémoire sensorielle par une couche sémantique afin d'améliorer l'interaction avec l'utilisateur. On peut ainsi imaginer que la mémoire fournisse des informations sur l'environnement ("pièce", "bureau", "route", "ruelle" ...) et sur le contenu de cet environnement ("feux de signalisation", "photocopieuse"...).

Dans le Chapitre 3, nous avons décrit les objectifs et développé des schémas de commande pour le suivi d'un chemin sensoriel pour deux types de robots : les robots mobiles à roues et les robots aériens de type quadrirotor. À partir de notre expérience sur ces deux cas particuliers, il serait intéressant de définir un formalisme générique pour tous les types de robots mobiles.

L'objectif de commande que nous avons défini consiste à amener le robot aux positions et aux orientations (dans le cas du quadrirotor, en lacet uniquement) correspondant aux prises de vue définissant le chemin sensoriel. Cependant, ces positions et orientations dépendent fortement du mode opératoire employé lors de l'apprentissage. Elles peuvent aboutir à un comportement peu naturel du robot et peu confortable pour le matériel et les passagers. On pourrait envisager de définir un schéma de commande plus souple afin de permettre l'ajout de contraintes additionnelles liées au type de robot, aux mécanismes du robot ainsi qu'à l'environnement. On pourrait, par exemple, s'intéresser aux contraintes d'évitements des butées des robots et d'optimisation de l'espace libre l'entourant. Comme nous l'avons évoqué dans la conclusion du chapitre 3, la planification directement dans l'image d'un chemin admissible pour le robot, prenant en compte ces contraintes, pourrait alors être employée. L'évitement d'obstacles pourrait également être introduit à ce niveau.

Enfin, nous nous sommes uniquement intéressés à la tâche de navigation pour les robots mobiles. Cependant, la stratégie proposée pourrait être étendue à d'autres tâches et à d'autres robots. On peut par exemple imaginer d'utiliser cette stratégie pour un bras manipulateur qui réalise une tâche de préhension d'un objet en utilisant des observations capteurs acquises lors d'une phase d'apprentissage.

Annexe A

Résultats de localisation initiale

Dans cette Annexe, nous présentons quelques exemples permettant de visualiser les résultats obtenus avec différentes approches de localisation initiale (voir Section 4.2).

Pour certains tests, toutes les méthodes considérées donnent un résultat relativement similaire. C'est le cas dans les exemples représentés Fig. A.1 pour une image d'ensemble *Almere* et Fig. A.2 pour une image de l'ensemble *UAV*.

Dans l'exemple Almere II, les résultats sont différents (voir Fig. A.3). Le résultat obtenu avec l'approche *PHLAC* est faux. On observe que la localisation obtenue par la méthode *Gonz* est similaire à l'image espérée à une rotation près. Les autres méthodes donnent des résultats satisfaisants. Les mesures de distance entre l'image à localiser et l'ensemble des images de la mémoire prises en compte sont représentées Fig. A.4 pour quelques descripteurs. On s'aperçoit que le descripteur *Sect* est peu discriminant dans cet exemple. Les stratégies de localisation globales présentent des courbes de distance relativement similaires entre elles ainsi que les stratégies locales.

Un exemple de localisation dans l'ensemble *Walk* est présenté Fig. A.5. L'image à localiser est l'image $\mathcal{I}_c$ (Fig. A.5(a)). Le résultat attendu est l'image $\mathcal{I}_{17}$ (Fig. A.5(g)). Les résultats des méthodes sont : $\mathcal{I}_{13}$ pour le *SIFT*, $\mathcal{I}_{14}$ pour le *SURF*, $\mathcal{I}_{15}$ pour *Gonz*, $\mathcal{I}_{17}$ pour la méthode *CubCorrHar* et $\mathcal{I}_{25}$ pour le *PHLAC*.

Nous considérons maintenant la localisation d'une image de l'ensemble *Walk* acquise en environnement d'extérieur avec la caméra dirigée vers le sol (voir Fig. A.6 (a)). Lors de l'acquisition de cette image, une partie de l'environnement était fortement illuminée tandis qu'une autre partie était située dans l'ombre de bâtiments. Des conditions similaires existaient lors de la phase de test. Les méthodes *Gonz* et *PHLAC* s'attachant plutôt à l'apparence globale de l'image donnent des résultats erronés tandis que les images trouvées par les autres méthodes sont relativement similaires à l'image test.

Nous analysons maintenant les résultats de la localisation d'une image de l'ensem-

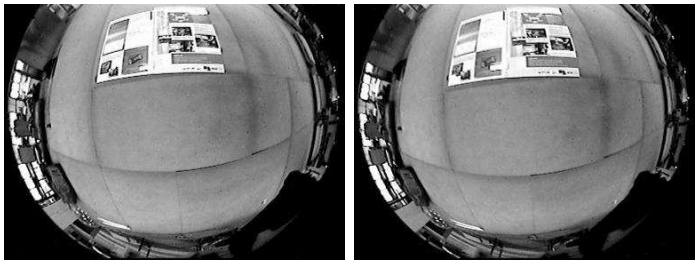
(a) $\mathcal{I}_c$ ($\mathcal{I}_{1066}$)

(b) $\mathcal{I}_{1021}$: *Triang*, *Cub*, *CorrHar*, *SectCorrHarr*, *CubCorrHar*

(c) $\mathcal{I}_{1061}$: *Sect*, *Gonz*, *HistoSect*, *SIFT*

(d) $\mathcal{I}_{1071}$: *PHLAC*, *SURF*

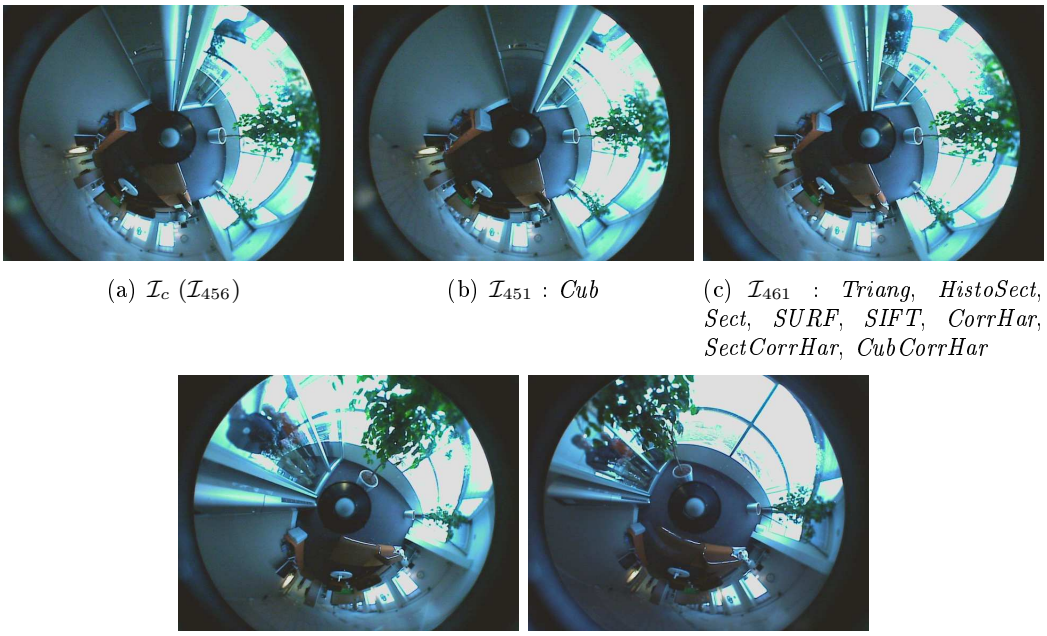
FIG. A.1 – Localisation Almere I : image à localiser $\mathcal{I}_c$ et résultats obtenus.



(a) $\mathcal{I}_c$ ($\mathcal{I}_{237}$)

(b) $\mathcal{I}_{235}$: *Triang*, *Cub*, *CorrHar*, *Sect*, *Gonz*, *HistoSect*, *SectCorrHarr*, *CubCorrHar*, *PHLAC*, *SURF*, *SIFT*

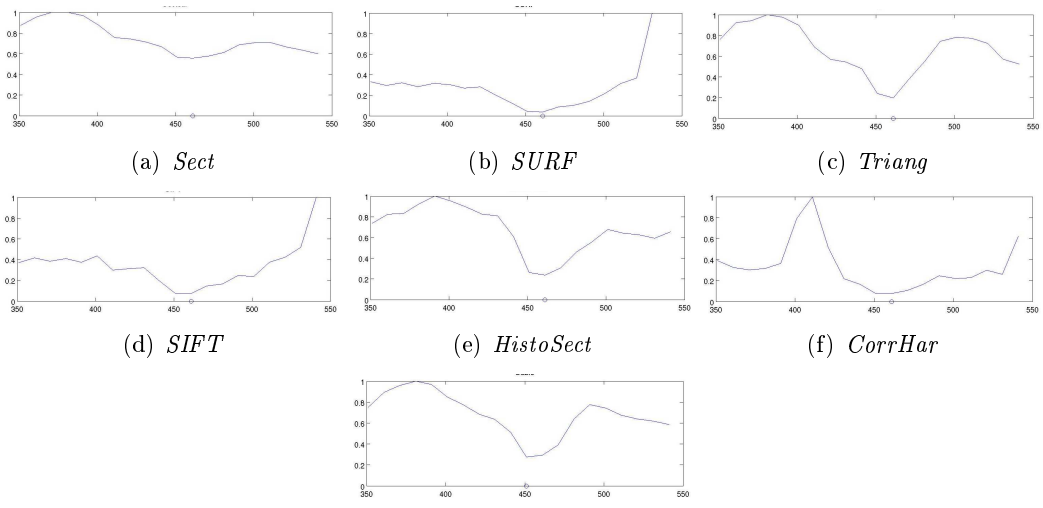
FIG. A.2 – Localisation UAV I : image à localiser $\mathcal{I}_c$ et résultat obtenu.



(a) $\mathcal{I}_c$ ($\mathcal{I}_{456}$) (b) $\mathcal{I}_{451}$: *Cub* (c) $\mathcal{I}_{461}$: *Triang*, *HistoSect*, *Sect*, *SURF*, *SIFT*, *CorrHar*, *SectCorrHar*, *CubCorrHar*

(d) $\mathcal{I}_{491}$: *Gonz* (e) $\mathcal{I}_{531}$: *PHLAC*

FIG. A.3 – Localisation Almere II (base *Almere*) : image à localiser $\mathcal{I}_c$ et résultats obtenus.

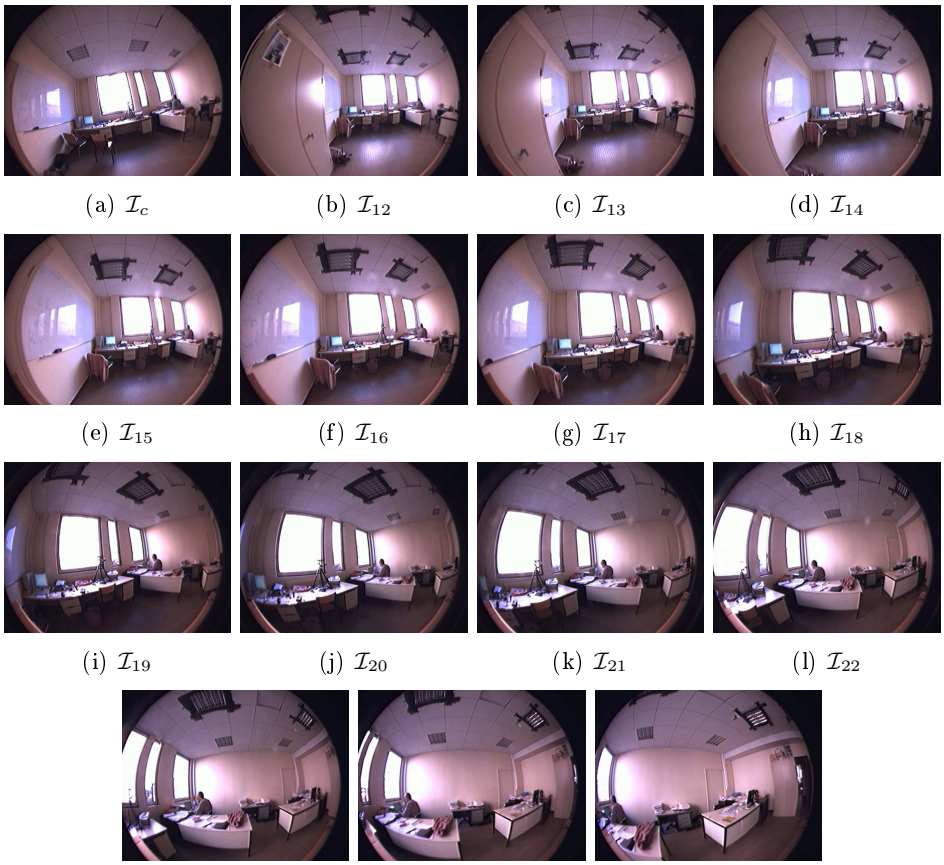


(a) *Sect* (b) *SURF* (c) *Triang*

(d) *SIFT* (e) *HistoSect* (f) *CorrHar*

(g) *Cub*

FIG. A.4 – Localisation Almere II : distance entre l'image courante $\mathcal{I}_c$ et les images de la base pour différents descripteurs.



(a) $\mathcal{I}_c$ (b) $\mathcal{I}_{12}$ (c) $\mathcal{I}_{13}$ (d) $\mathcal{I}_{14}$

(e) $\mathcal{I}_{15}$ (f) $\mathcal{I}_{16}$ (g) $\mathcal{I}_{17}$ (h) $\mathcal{I}_{18}$

(i) $\mathcal{I}_{19}$ (j) $\mathcal{I}_{20}$ (k) $\mathcal{I}_{21}$ (l) $\mathcal{I}_{22}$

(m) $\mathcal{I}_{23}$ (n) $\mathcal{I}_{24}$ (o) $\mathcal{I}_{25}$

FIG. A.5 – Image à localiser $\mathcal{I}_c$ et sous ensemble des images de la base *Walk* ($\mathcal{I}_{12} \ldots \mathcal{I}_{22}$).

(a) $\mathcal{I}_c$ (b) $\mathcal{I}_{80}$: *PHLAC* (c) $\mathcal{I}_{124}$: *Gonz*

(d) $\mathcal{I}_{198}$: *Triang*, *HistoSect*, *Cub*, *SURF*, *SIFT* (e) $\mathcal{I}_{200}$: *Sect*, *CorrHar*, *SectCorrHar*, *CubCorrHar*

FIG. A.6 – Localisation Walk I (base *Walk*) : image à localiser $\mathcal{I}_c$ et résultats obtenus.

ble *Walk* acquise dans un environnement d'extérieur avec la caméra dirigée vers l'avant (voir Fig. A.7 (a)). Dans cet exemple, les conditions d'illumination ont relativement changé entre la phase d'apprentissage et la phase de test. L'approche *SURF* donne ici le meilleur résultat et on observe que le contenu des images trouvées par les approches *Cub* et *Sect* est différent de celui de l'image test.

Nous étudions enfin les résultats de la localisation pour les images acquises dans un environnement peu structuré. Lors de l'apprentissage de la base *Walk*, une courte séquence a été acquise dans un environnement naturel. On observe Fig. A.8 que les résultats ne sont pas satisfaisants. On peut voir sur les courbes de distance représentées Fig. A.9 que les méthodes sont souvent peu discriminantes. Notre stratégie, comme les autres méthodes considérées, est donc peu adaptée à ce type d'environnement. Afin d'améliorer les résultats, il serait nécessaire de prendre en compte d'autres informations contenues dans les images telles que la couleur ou la texture.

(a) $\mathcal{I}_c$

(b) $\mathcal{I}_{154}$: *PHLAC*

(c) $\mathcal{I}_{155}$: *Triang, Sect*

(d) $\mathcal{I}_{159}$: *Gonz*

(e) $\mathcal{I}_{162}$: *SectCorrHar*

(f) $\mathcal{I}_{181}$: *SURF*

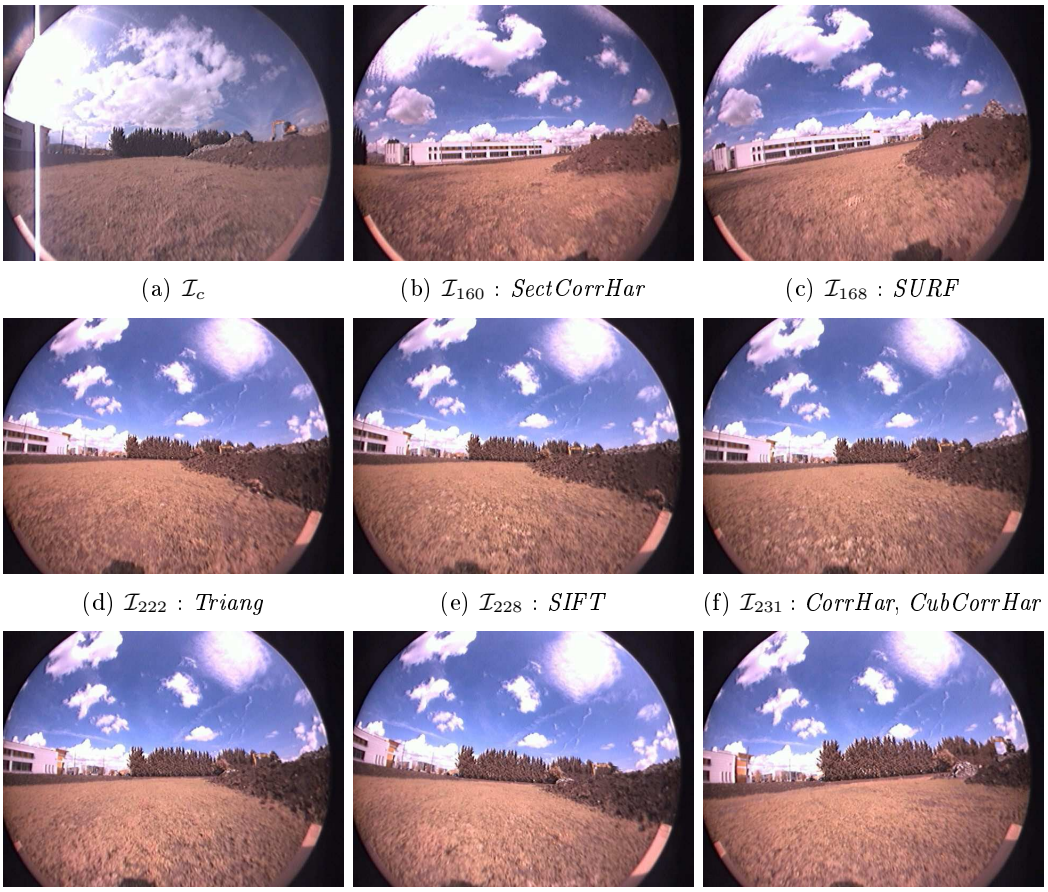
(g) $\mathcal{I}_{195}$: *CorrHar, CubCorrHar*

(h) $\mathcal{I}_{197}$: *SIFT*

(i) $\mathcal{I}_{237}$: *Cub*

(j) $\mathcal{I}_{248}$: *HistoSect*

FIG. A.7 – Localisation Walk II : image à localiser $\mathcal{I}_c$ et résultats obtenus.

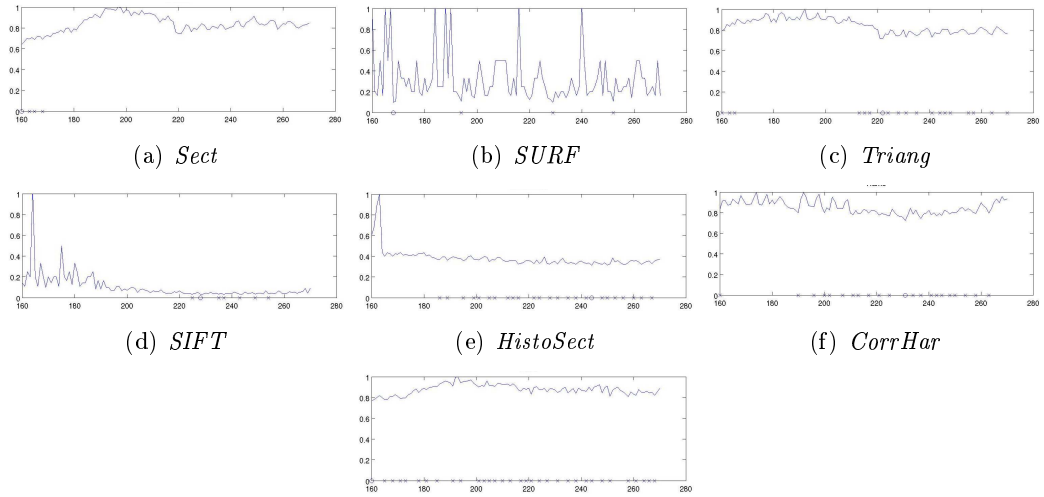


(a) $\mathcal{I}_c$ (b) $\mathcal{I}_{160}$: *SectCorrHar* (c) $\mathcal{I}_{168}$: *SURF*

(d) $\mathcal{I}_{222}$: *Triang* (e) $\mathcal{I}_{228}$: *SIFT* (f) $\mathcal{I}_{231}$: *CorrHar*, *CubCorrHar*

(g) $\mathcal{I}_{244}$: *HistoSect* (h) $\mathcal{I}_{249}$: *Gonz* (i) $\mathcal{I}_{268}$: *PHLAC*

FIG. A.8 – Localisation Walk III : image à localiser $\mathcal{I}_c$ et résultats obtenus.



(a) *Sect* (b) *SURF* (c) *Triang*

(d) *SIFT* (e) *HistoSect* (f) *CorrHar*

(g) *Cub*

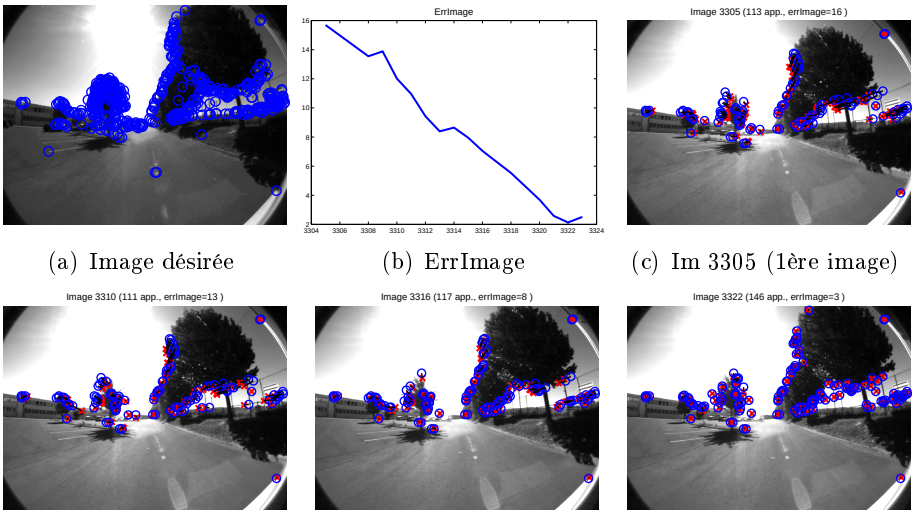
FIG. A.9 – Localisation Walk III : distance entre l'image courante $\mathcal{I}_c$ et les images de la base pour différents descripteurs.

Annexe B

Comportement dans l'image

Nous considérons la seconde expérimentation détaillée Section 5.5.4. Le comportement des points dans les 16 images acquises entre deux images clés successives est analysé (voir Fig. B.1). Les positions des points à atteindre dans l'image désirée (voir Fig. B.1 (a)) sont marquées par des cercles tandis que les positions des points appareillés dans les images acquises lors de la phase autonome sont marquées par des croix (voir Fig. B.1 (c), (d), (e), (f)). La distance moyenne entre les points appareillés ErrImage est représentée Fig. B.1 (b). L'erreur dans l'image est initialement de 16 pixels (Im 3305) et décroit jusqu'à atteindre 3 pixels à l'image Im 3322.

Nous nous intéressons maintenant aux points d'intérêts de cette image de référence appareillés avec des points des images Im 3305 à Im 3322. On rappelle que 500 points ont été détectés dans l'image désirée. Sur ces 500 points, seuls 200 ont été appareillés avec les points des 16 images acquises lors de la phase autonome (voir Fig. B.2), dont la moitié avec les points d'au moins 10 images. Il serait intéressant d'étudier plus en détail ce phénomène sur les différentes images de la mémoire à partir des phases de suivi autonome. Ainsi, si des points des images de la mémoire ne sont jamais utilisés (que ce soit pour la localisation initiale ou pour l'appariement robuste en vu de la commande), il est envisageable de les supprimer de la mémoire. Cela permettrait alors de diminuer sa taille. De plus, dans le cas présenté, les points sont situés principalement sur le feuillage des arbres. Il serait intéressant d'étudier le comportement du suivi lorsque ces arbres ont perdu leur feuillage.



(a) Image désirée (b) ErrImage (c) Im 3305 (1ère image)

(d) Im 3310 (e) Im 3316 (f) Im 3322

FIG. B.1 – Image de la mémoire à rejoindre, erreurs dans l'image sur les images acquises jusqu'à rejoindre cette image et certaines de ces images. Le centre des cercles représente la position du point dans l'image désirée tandis que le milieu d'une croix sa position dans l'image courante.

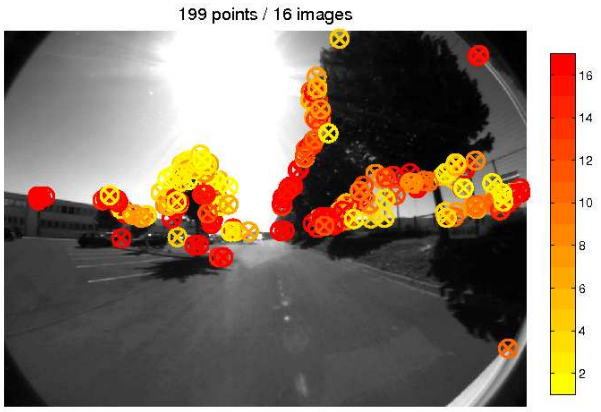


FIG. B.2 – Points de l'image désirée appareillés avec des images courantes et nombre de fois que ces points ont été utilisés.

Bibliographie

[Alami 98] R. Alami, R. Chatila, S. Fleury, M. Ghallab & F. Ingrand. *An architecture for autonomy.* The International Journal of Robotics Research, Special Issue on Integrated Architectures for Robot Control and Programming, vol. 17, no. 4, pp. 315–337, 1998.

[Andreasson 05] H. Andreasson, A. Treptow & T. Duckett. *Localization for mobile robots using panoramic vision, local features and particle filter.* IEEE International Conference on Robotics and Automation, ICRA'05, pp. 3348–3353, Barcelone, Espagne, Avril 2005.

[Angeli 08a] A. Angeli. *Détection visuelle de fermeture de boucle : applications à la localisation et cartographie simultanées.* Thèse de doctorat, Université Paris VI, 2008.

[Angeli 08b] A. Angeli, D. Filliat, S. Doncieux & J.-A. Meyer. *Fast and Incremental Method for Loop-Closure Detection using Bags of Visual Words.* IEEE Transactions on Robotics, Special issue on Visual SLAM, vol. 24, pp. 1027–1037, 2008.

[Arbib 81] M.A. Arbib. Handbook of Physiology – The Nervous System II. Motor Control, chapitre Perceptual structures and distributed motor control. American Physiology Society, 1981.

[Argyros 01] A. Argyros, K. Bekris & S. Orphanoudakis. *Robot Homing based on Corner Tracking in a sequence of Panoramic Images.* IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), vol. 2, pp. 3–10, Hawaii, USA, Décembre 2001.

[Argyros 05] A. Argyros, K.Bekris & S. Orphanoudakis. *Robot Homing by Exploiting Panoramic Vision.* Journal of Autonomous Robots, vol. 19, pp. 7–25, Juillet 2005.

[Arleo 00] A. Arleo & W. Gerstner. *Spatial cognition and neuro-mimetic navigation : a model of hippocampal place cell activity.* Biological Cybernetics, Special Issue on Navigation in Biological and Artificial Systems, vol. 83, pp. 287–299, Septembre 2000.

[Astolfi 96] A. Astolfi. *Discontinuous control of nonholonomic systems.* Systems & Control Lett., vol. 27, pp. 37–45, 1996.

[Atkinson 68] R.C. Atkinson & R.M. Shiffrin. The psychology of learning and motivation, chapitre Human memory : a proposed system and its control processes. New York : Academic Press, spence, k. w. and spence, j. t. edition, 1968.

[Baddeley 74] A.D. Baddeley & G. Hitch. *Working memory.* The Psychology of Learning and Motivation, pp. 48–79, 1974.

[Bailey 06] T. Bailey & H. Durrant-Whyte. *Simultaneous Localisation and Mapping (SLAM) : Part II State of the Art.* Robotics and Automation Magazine, Septembre 2006.

[Baker 98] S. Baker & S.K. Nayar. *A Theory of Catadioptric Image Formation.* International Conference on Computer Vision, ICCV'98, pp. 35–42, Bombay, India, Janvier 1998.

[Baker 99] S. Baker & S.K. Nayar. *A theory of single-viewpoint catadioptric image formation.* International Journal of Computer Vision, vol. 35, no. 2, pp. 1–22, November 1999.

[Bakstein 02] H. Bakstein & T. Pajdla. *Panoramic Mosaicing with a* 180° *Field of View Lens.* IEEE Workshop on Omnidirectional Vision, pp. 60–67, Los Alamitos, USA, 2002.

[Barreto 02] J. Barreto & H. Araujo. *Geometric Properties of Central Catadioptric Line Images.* 7th European Conference on Computer Vision, ECCV'2002, pp. 117–120, Copenhague, Suède, Mai 2002.

[Barreto 03] J.P. Barreto. *General Central Projection Systems, modeling, calibration and visual servoing.* Thèse de doctorat, University of Coimbra, Avril 2003.

[Barreto 06] J.P. Barreto. *A unifying geometric representation for central projection systems.* Computer Vision and Image Understanding, vol. 103, no. 3, pp. 208–217, Septembre 2006. Special issue on omnidirectional vision and camera networks.

[Basu 95] A. Basu & S. Licardie. *Alternative models for fish-eye lenses.* Pattern Recognition Letters, vol. 16, no. 4, pp. 433–441, 1995.

[Bay 08] H. Bay, T. Tuytelaars & L. Van Gool. *SURF : Speeded up robust features.* Computer Vision and Image Understanding (CVIU), vol. 110, no. 3, pp. 346–359, 2008.

[Becerra 08] H.M. Becerra & C. Sagues. *A Sliding Mode Control Law for Epipolar Visual Servoing of Differential-Drive Robots.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'08, pp. 3058–3063, Nice, France, Septembre 2008.

[Beevers 05] K.R. Beevers & W. Huang. *Loop closing in topological maps.* IEEE International Conference on Robotics and Automation, ICRA'05, pp. 4378–4383, Barcelone, Espagne, Avril 2005.

[Benallegue 07] A. Benallegue, A. Mokhtari & L. Fridman. *High order sliding mode observer for a quadrotor UAV.* International Journal of Robuste and Nonlinear Control, vol. 18, no. 4–5, pp. 427–440, Mai 2007.

[Benosman 00] R. Benosman & S. Kang. Panoramic vision. Springer Verlag ISBN 0-387-95111-3, 2000.

[Bertrand 07] S. Bertrand, T. Hamel & H. Piet-Lahanier. *Trajectory Tracking of an Unmanned Aerial Vehicle Model using Partial State Feedback.* European Control Conference, pp. 307–314, Kos, Grèce, Juillet 2007.

[Blaer 02] P. Blaer & P.K. Allen. *Topological mobile robot localization using fast vision techniques.* IEEE International Conference on Robotics and Automation, ICRA'02, pp. 1031–1036, Washington, USA, Mai 2002.

[Blanc 05] G. Blanc, Y. Mezouar & P. Martinet. *Indoor navigation of a wheeled mobile robot along visual routes.* IEEE International Conference on Robotics and Automation, ICRA'05, pp. 3365–3370, Barcelone, Espagne, Avril 2005.

[Bonnifait 08] Ph. Bonnifait, M. Jabbour & V. Cherfaoui. *Autonomous Navigation in Urban Areas using GIS-Managed Information.* International Journal of Vehicle Autonomous Systems (IJVAS), vol. 6, no. 1/2, pp. 83–103, Décembre 2008. Special Issue on Advances in Autonomous Vehicles and Intelligent Transportation.

[Booij 07] O. Booij, B. Terwijn, Z. Zivkovic & B. Kröse. *Navigation using an appearance based topological map.* IEEE International Conference on Robotics and Automation, ICRA'07, pp. 3927–3932, Rome, Italie, Avril 2007.

[Bouabdallah 04a] S. Bouabdallah, A.Noth & R.Siegwart. *PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'04, vol. 3, pp. 2451–2456, Sendai, Japon, Septembre 2004.

[Bouabdallah 04b] S. Bouabdallah, P. Murrieri & R. Siegwart. *Design and control of an indoor micro quadrotor.* IEEE International Conference on Robotics and Automation, ICRA'04, vol. 5, pp. 4393–4398, Nouvelle Orléans, USA, Avril 2004.

[Bouabdallah 05] S. Bouabdallah & R. Siegwart. *Backstepping and Sliding-mode Techniques Applied to an Indoor Micro Quadrotor.* IEEE International Conference on Robotics and Automation, ICRA'05, pp. 2247–2252, Barcelone, Espagne, Avril 2005.

[Bourquardez 09] O. Bourquardez, R. Mahony, N. Guénard, F. Chaumette, T. Hamel & L. Eck. *Image-based visual servo control of the*

translation kinematics of a quadrotor aera vehicle. IEEE Transactions on Robotics, vol. 25(3), pp. 743–749, Juin 2009.

[Braitenberg 84] V. Braitenberg. Vehicles. MIT Press, Cambridge, 1984.

[Brame 96] J.-L. Brame, J. Alizon & J. Gallice. *Device for exposure control of a CCD camera.* Brevet, Juillet 1996. Patent no EP07243558.

[Briggs 06] A. Briggs, Y. Li, D. Scharstein & M. Wilder. *Robot navigation using 1D panoramic images.* IEEE International Conference on Robotics and Automation, ICRA'06, pp. 2679–2685, Orlando, Floride, Mai 2006.

[Brockett 83] R.W. Brockett. Asymptotic stability and feedback stabilization. differential geometric control theory. Birkhauser, 1983.

[Broggi 99] A. Broggi, M. Bertozzi, A. Fascioli, C.G. Lo Bianco & A. Piazzi. *The ARGO autonomous vehicle's vision and control systems.* International Journal of Intelligent Control and Systems, vol. 3, no. 4, pp. 409–441, 1999.

[Brooks 86] R.A. Brooks. *A Robust Layered Control System for a Mobile Robot.* IEEE Journal of Robotics and Automation, vol. 2, pp. 14–23, Mars 1986.

[Buschka 04] P. Buschka & A. Saffiotti. *Some Notes on the Use of Hybrid Maps for Mobile Robots.* 8th International Conference on Intelligent Autonomous Systems (IAS), pp. 547–556, Amsterdam, NL, 2004.

[Cadenat 99] V. Cadenat. *Commande référencée multi-capteurs pour la navigation d'un robot mobile.* Thèse de doctorat, Université Paul Sabatier, Toulouse, France, DÃ¯Â¿Â½embre 1999.

[Cartwright 83] B.A. Cartwright & T.S. Collett. *Landmark learning in bees.* Journal of Comparative Physiology, pp. 521–543, 1983.

[Castillo 04] P. Castillo, A. Dzul & R. Lozano. *Real-time stabilization and tracking of a four rotor mini aircraft.* IEEE Transactions on Control Systems Technology, vol. 12(4), pp. 116–129, Juillet 2004.

[Cervera 04] E. Cervera & P. Martinet. *A tutorial on Advanced Visual Servoing.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'04, Sendai, Japon, Septembre 2004. http ://www.robot.uji.es/EURON/visualservoing/tutorial/.

[Charron 06] C. Charron, O. Labbani-Igbida & E. Mouaddib. *On Building Omnidirectional Image Signatures Using Haar Invariant Features : Application to the Localization of Robots.* Advanced Concepts for Intelligent Vision Systems (ACIVS), pp. 1099–1110, Antwerp, Belgique, Septembre 2006.

[Chatila 85] R. Chatila & J.P. Laumond. *Position referencing and consistent world modeling for mobile robots.* IEEE International Conference on Robotics and Automation, ICRA'85, vol. 2, pp. 138–145, St. Louis, Missouri, USA, 1985.

[Chaumette 06] F. Chaumette & S. Hutchinson. *Visual Servo Control, Part I : Basic Approaches.* IEEE Robotics and Automation Magazine, vol. 13, no. 4, pp. 82–90, Décembre 2006.

[Chaumette 07] F. Chaumette & S. Hutchinson. *Visual Servo Control, Part II : Advanced Approaches.* IEEE Robotics and Automation Magazine, vol. 14, no. 1, pp. 109–118, Mars 2007.

[Chen 06] Z. Chen & S. Birchfield. *Qualitative Vision-Based Mobile Robot Navigation.* IEEE International Conference on Robotics and Automation, ICRA'06, pp. 2686–2692, Orlando, Floride, 2006.

[Cherkassky 96] B. Cherkassky, A. Goldberg & T. Radzik. *Shortest paths algorithms : theory and experimental evaluation.* Mathematical Programming, vol. 73, no. 2, pp. 129–174, Mai 1996.

[Cherubini 09] A. Cherubini, M. Colafrancesco, G. Oriolo, L. Freda & F. Chaumette. *Comparing appearance-based controllers for nonholonomic navigation from a visual memory.* ICRA'09 Workshop on safe navigation in open and dynamic environments : application to autonomous vehicles, Kobe, Japon, Mai 2009.

[Chitrakaran 06] V. Chitrakaran, D. Dawson, H. Kannan & M. Feemster. *Assisted Autonomous Path Following for Unmanned Aerial Vehicles.* Rapport technique, Clemson University CRB Technical Report, CU/CRB/2/27/06, Mars 2006.

[Choset 95] H. Choset & J. Burdick. *Sensor Based Planning, Part I : The Generalized Voronoi Graph.* IEEE International Conference on Robotics and Automation, ICRA'95, vol. 2, pp. 1649–1655, Nagoya, Japon, Mai 1995.

[Choset 01] H. Choset & K. Nagatani. *Topological simultaneous localization and mapping (SLAM) : toward exact localization without explicit localization.* IEEE Transactions on Robotics and Automation, vol. 17, no. 2, pp. 125–137, Avril 2001.

[Cowan 99] N. Cowan. Models of working memory : Mecanisms of active maintenance and executive control, chapitre An Embedded-Processes Model of Working Memory. Cambridge : Cambridge University Press, A. Miyake and P. Shah edition, 1999.

[Data Set 06] Data Set. *Workshop- From Sensors to Human Spatial Concepts- FS2HSC-data.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'06, Octobre 2006. http ://staff.science.uva.nl/˜ zivkovic/FS2HSC/dataset.html. Dernière visite : juin 2007.

[Davison 03] A.J. Davison. *Real-time simultaneous localisation and mapping with a single camera.* International Conference on Computer Vision, ICCV, Nice, France, Octobre 2003.

[De Luca 00] A. De Luca, G. Oriolo & M. Vendittelli. *Stabilization of the unicycle via dynamic feedback linearization.* 6th IFAC Symposium on Robot Control, pp. 397–402, Vienne, Autriche, Septembre 2000.

[Demonceaux 06] C. Demonceaux & P. Vasseur. *Mesure d'Attitude pour les Drones par Vision Catadioptrique Centrale.* 15ème Congrès de Reconnaissance des Formes et Intelligence Artificielle, RFIA 2006, Tours, France, Janvier 2006.

[DeSouza 02] G. N. DeSouza & A. C. Kak. *Vision for Mobile Robot Navigation : A Survey.* IEEE transactions on pattern analysis and machine intelligence, vol. 24, no. 2, pp. 237–267, Février 2002.

[Devernay 01] F. Devernay & O. Faugeras. *Straight lines have to be straight.* Machine Vision and Applications, vol. 13, no. 1, pp. 14–24, Août 2001.

[Diosi 07] A. Diosi, S. Šegvić, A. Remazeilles & F. Chaumette. *Experimental evaluation of an urban visual path following framework.* 6th IFAC Symposium on Intelligent Autonomous Vehicles, IAV'07, Toulouse, France, Septembre 2007.

[Durrant-Whyte 06] H. Durrant-Whyte & T. Bailey. *Simultaneous Localisation and Mapping (SLAM) : Part I The Essential Algorithms.* Robotics and Automation Magazine, Juin 2006.

[Elfes 89] A. Elfes. *Using occupancy grids for mobile robot perception and navigation.* Computer, vol. 22, no. 6, pp. 46–57, Juin 1989.

[Faugeras 88] O. Faugeras & F. Lustman. *Motion and structure from motion in a piecewise planar environment.* International Journal of Pattern Recognition and Artificial Intelligence, vol. 2, no. 3, pp. 485–508, 1988.

[Fermuller 00] C. Fermuller, Y. Aloimonos, P. Baker, R. Pless, J. Neumann & B. Stuart. *Multi-camera networks : Eyes from eyes.* IEEE Workshop on Omnidirectional Vision, pp. 11–18, Hilton Head Island, USA, Juin 2000.

[Filliat 01] David Filliat. *Cartographie et estimation globale de la position pour un robot mobile autonome.* Thèse de doctorat, Université Paris 6, 2001.

[Filliat 03] D. Filliat & J.A. Meyer. *Map-based navigation in Mobile robots - II. A review of map-learning and path-planing strategies.* Journal of Cognitive Systems Research, vol. 4, no. 4, pp. 283–317, 2003.

[Filliat 07] D. Filliat. *A visual bag of words method for interactive qualitative localization and mapping.* IEEE International Conference on Robotics and Automation, ICRA'07, pp. 3921–3926, Rome, Italie, Avril 2007.

[Fischler 81] M. Fischler & R. Bolles. *Random Sample Consensus : A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography.* Communication of the ACM, vol. 24, pp. 381–395, Juin 1981.

[Fitzgibbon 01] A. Fitzgibbon. *Simultaneous Linear Estimation of Multiple-view Geometry and Lens Distortion.* IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2001, vol. 1, pp. I–125–I–132, Hawaii, USA, Décembre 2001.

[Fleck 94] M.M. Fleck. *Perspective Projection : the Wrong Imaging Model.* Rapport technique, Computer Science. University of Iowa. USA, 1994. TR 95–01.

[Folio 07] D. Folio & V. Cadenat. *A new controller to perform safe vision-based navigation tasks amidst possibly occluding obstacles.* European Control Conference, ECC'07, Kos, Greece, JUL 2007.

[Fraichard 05] T. Fraichard. *Cybercar : l'alternative à la voiture particulière.* Navigation (Paris), vol. 53, pp. 53–74, Janvier 2005.

[Galindo 05] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J.A. Fernandez-Madrigal & J. Gonzalez. *Multi-Hierarchical Semantic Maps for Mobile Robotics.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'05, pp. 2278–2283, Edmonton, Canada, Août 2005.

[Gaspar 00] J. Gaspar, N. Winters & J. Santos-Victor. *Vision-based Navigation and Environmental Representations with an Omnidirectional Camera.* IEEE Transaction on Robotics and Automation, vol. 16, pp. 890–898, Décembre 2000.

[Gaussier 97] P. Gaussier, C. Joulain, S. Zrehen, J. Banquet & A. Revel. *Visual navigation in an open environment without map.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'97, pp. 545–550, Grenoble, France, 1997.

[Geyer 00] C. Geyer & K. Daniilidis. *A unifying theory for central panoramic systems and practical implications.* European Conference on Computer Vision, vol. 29 (3), pp. 159–179, Dublin, Irlande, Mai 2000.

[Geyer 03] C. Geyer. *Catadioptric Projective Geometry : theory and applications.* Thèse de doctorat, Université de Pennsylvanie, 2003.

[Giovannangeli 06] C. Giovannangeli & Ph Gaussier. *Robust Mapless Outdoor Vision-based Navigation.* IEEE/RSJ International Conference

on Intelligent Robots and Systems, IROS'06, pp. 3293–3300, Pékin, Chine, Octobre 2006.

[Goedemé 05] T. Goedemé, T. Tuytelaars, G. Vanacker, M. Nuttin, L. Van Gool & L. Van Gool. *Feature based omnidirectional sparse visual path following.* IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1806–1811, Edmonton, Canada, Août 2005.

[Golledge 01] R.G. Golledge & T. Gärling. Spatial behaviour in transportation modeling and planning, chapitre 3 for the *Transportation and Engineering Handbook.* K. Goulias, Eds., 2001.

[Gonzalez-Barbosa 02] J. Gonzalez-Barbosa & S. Lacroix. *Rover localization in natural environments by indexing panoramic images.* IEEE International Conference on Robotics and Automation, ICRA'02, vol. 2, pp. 1365–1370, Washington DC, USA, Mai 2002.

[Gourichon 04] S. Gourichon. *Utilisation d'un compas visuel pour la navigation d'un robot mobile.* Thèse de doctorat, LIP6/AnimatLab, Université Pierre et Marie Curie, Paris, France, 2004.

[Guénard 04] N. Guénard, T. Hamel & V. Moreau. *Modélisation et élaboration de commande de stabilisation de vitesse et de correction d'assiette pour un drone de type X4-Flyer.* Conférence Internationale Francophone d'Automatique, CIFA'2004, Douz, Tunisie, Novembre 2004.

[Guénard 06] N. Guénard. *Optimisation et implémentation de lois de commande embarquées pour la téléopération de micro drones aériens "X4-flyer".* Thèse de doctorat, Université Nice Sophia Antipolis, 2006.

[Guénard 07] N. Guénard, T. Hamel & R. Mahony. *A practical Visual Servo Control for a Unmanned Aerial.* IEEE International Conference on Robotics and Automation, ICRA'07, pp. 1342–1348, Rome, Italie, Avril 2007.

[Guénard 08] N. Guénard, V. Moreau, T. Hamel & R. Mahony. *Synthèse d'un contrôleur permettant la stabilisation de vitesse d'un drone de type X4-Flyer via la correction d'assiette.* Journal européen des systèmes automatisés (RS-JESA), vol. 42, no. 1, pp. 117–138, 2008.

[Hamel 02a] T. Hamel & R. Mahony. *Visual servoing of an underactuated dynamic rigid-body system : An image based approach.* IEEE Transactions on Robotics and Automation, vol. 18, no. 2, pp. 187–198, Avril 2002.

[Hamel 02b] T. Hamel, R. Mahony, R. Lozano & J.-N. Ostrowski. *Dynamic Modelling and Configuration Stabilization for an X4-*

flyer. 15th International Federation of Automatic Control symposium, IFAC'2002, Barcelone, Espagne, Juillet 2002.

[Hamel 06a] T. Hamel, L. Eck, F. Chaumette & A. Chriette. *Rapport final du projet ROBVOLINT (ROBot VOLant d'INTérieur).* Journées ROBEA 2006, 2006.

[Hamel 06b] T. Hamel & R. Mahony. *Attitude estimation on SO(3) based on direct inertial measurements.* IEEE International Conference on Robotics and Automation, ICRA'06, pp. 2170–2175, Orlando, Floride, Mai 2006.

[Harris 88] C. Harris & M. Stephens. *A combined corner and edge detector.* The Fourth Alvey Vision Conference, pp. 147–151, Manchester, UK, 1988.

[Hartley 00] R. Hartley & A. Zisserman. Multiple view geometry in computer vision. Cambridge University Press, ISBN : 0521623049, 2000.

[Hong 91] J. Hong, X. Tan, B. Pinnete, R. Weiss & E. Riseman. *Image-Based homing.* IEEE International Conference on Robotic and Automation, vol. 1, pp. 620–625, Sacramento, USA, Avril 1991.

[Hua] M.-D. Hua, T. Hamel, P. Morin & C. Samson. *A Control Approach for Thrust-Propelled Underactuated Vehicles and its application to VTOL drones.* IEEE Transactions on Automatic Control. To appear.

[Hutchinson 96] S. Hutchinson, G. Hager & P. Corke. *A tutorial on visual servo control.* IEEE Transactions on Robotics and Automation, vol. 12, no. 5, pp. 651–670, 1996.

[Ieng 03] S.H. Ieng, R. Benosman & J. Devars. *An Efficient Dynamic Multi-Angular Feature Points Matcher for Catadioptric Views.* Workshop OmniVis'03, in conjunction with Computer Vision and Pattern Recognition (CVPR), vol. 07, pp. 75, Wisconsin, USA, Juin 2003.

[Kannala 06] J. Kannala & S. Brandt. *A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 8, pp. 1335–1340, Août 2006.

[Kim 03] H.J. Kim, D.H. Shim & S. Sastry. *A Flight Control System for Aerial Robots : Algorithms and Experiments.* Control Engineering Practise, vol. 11, no. 12, pp. 1389–1400, Mai 2003.

[Kingslake 89] R. Kingslake. A history of the photographic lens. Academic Press, San Diego, 1989. 334 p., ISBN 0-12-408640-3.

[Kirigin 05] I. Kirigin & S. Singh. *Bearings based robot homing with robust landmark matching and limited horizon view.* Rapport technique CMU-RI-TR-05-02, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Janvier 2005.

[Kortenkamp 94] D. Kortenkamp & T. Weymouth. *Topological mapping for mobile robots using a combination of sonar and vision sensing.* Twelfth National Conference on Artificial Intelligence (AAAI-94), pp. 979–984, Seattle, USA, Juillet 1994.

[Kuipers 91] B. Kuipers & Y.-T. Byun. *A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations.* Robotics and autonomous systems, vol. 8, no. 1–2, pp. 47–63, 1991.

[Kuipers 00] B. Kuipers. *The Spatial Semantic Hierarchy.* Artificial Intelligence, vol. 119, pp. 191–233, 2000.

[Kumler 00] J. Kumler & M. Bauer. *Fisheye lens designs and their relative performance.* SPIE, vol. 4093, pp. 360–369, 2000.

[Latombe 91] J. C. Latombe. Robot motion planning. Kluwer Academic Publishers, ISBN 3-540-76219-1, 1991.

[Lemaire 07] T. Lemaire, C. Berger, I.K. Jung & S. Lacroix. *Vision-based SLAM : Stereo and Monocular Approaches.* International Journal of Computer Vision, vol. 74, no. 3, pp. 343 – 364, Février 2007.

[Li 05] H. Li & R. Hartley. *An Easy Non-iterative Method for Correcting Lens Distortion from Nine Point Correspondences.* Workshop on Omnidirectional Vision, Camera Networks and Non-classical cameras, OMNIVIS'05, Pékin, Chine, 2005.

[Linåker 04] F. Linåker & M. Ishikawa. *Rotation invariant features from omnidirectional camera images using a polar higher-order local autocorrelation feature extractor.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'04, vol. 4, pp. 4026–4031, Sendai, Japon, Septembre 2004.

[Lowe 04] D.G. Lowe. *Distinctive image features from scale-invariant keypoints.* International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, Novembre 2004.

[Lozano 05] R. Lozano, P. Castillo, S. Salazar & D. Lara. *Stabilisation de véhicules aériens à décollage vertical : théorie et application.* Journées Nationales de la Recherche en Robotiques, JNRR'05, Guidel, France, Octobre 2005.

[Luria 85] A. Luria. Les fonctions corticales supérieures de l'homme. Presses universitaires de France, Juillet 1985.

[Maes 90] P. Maes. *A bottom-up mechanism for behavior selection in an artificial creature.* Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats, pp. 238–246, Cambridge, MA, USA, 1990. MIT Press.

[Mahony 02] R. Mahony & T. Hamel. *Robust Trajectory Tracking for a Scale Model Autonomous Helicopter.* International Journal of Nonlinear and Robust Control, pp. 1035–1059, 2002.

[Malis 00] E. Malis & F. Chaumette. *2 1/2 D Visual Servoing with respect to unknown objects through a new estimation scheme of camera displacement.* International Journal of Computer Vision, vol. 37(1), pp. 79–97, Juin 2000.

[Matarić 92] M. Matarić. *Integration of Representation Into Goal-Driven Behavior-Based Robots.* IEEE Transactions on Robotics and Automation, vol. 8(3), pp. 304–312, Juin 1992.

[Matsumoto 96] Y. Matsumoto, M. Inaba & H. Inoue. *Visual Navigation using View-Sequenced Route Representation.* IEEE International Conference on Robotics and Automation, ICRA'96, vol. 1, pp. 83–88, Minneapolis, Minnesota, USA, Avril 1996.

[Matsumoto 99] Y. Matsumoto, K. Ikeda, M. Inaba & H. Inoue. *Visual Navigation using Omnidirectional View Sequence.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'99, vol. 1, pp. 317–322, Kyongju, Corée, Octobre 1999.

[Mauthner 06] T. Mauthner, F. Fraundorfer & H. Bischof. *Region matching for omnidirectional images using virtual camera planes.* 11th Computer Vision Winter Workshop, pp. 93–98, Telc, République Tchèque, Février 2006.

[Mei 07] C. Mei & P. Rives. *Single View Point Omnidirectional Camera Calibration from Planar Grids.* IEEE International Conference on Robotics and Automation, ICRA'07, pp. 3945–3950, Rome, Italie, Avril 2007.

[Menegatti 04] E. Menegatti, T. Maeda & H. Ishiguro. *Image-based memory for robot navigation using properties of omnidirectional images.* Robotics and Autonomous Systems, Elsevier, vol. 47, no. 4, pp. 251–267, Septembre 2004.

[Metni 05] N. Metni, T. Hamel & F. Derkx. *Visual Tracking Control of Aerial Robotic Systems with Adaptive Depth Estimation.* 44th IEEE Conference on Decision and Control and European Control Conference (ECC), pp. 6078–6084, Seville, Espagne, Décembre 2005.

[Mezouar 01] Y. Mezouar. *Planification de trajectoires pour l'asservissement visuel.* Thèse de doctorat, Université de Rennes 1, 2001.

[Miller 56] G.A. Miller. *The Magical Number Seven, Plus or Minus Two : Some limits on our Capacity for Processing Information.* The Psychological Review, vol. 63, no. 2, pp. 81–97, 1956.

[Murillo 07a] A. C. Murillo, C. Sagüés, J. J. Guerrero, T. Goedemé, T. Tuytelaars & L. Van Gool. *From Omnidirectional Images to Hierarchical Localization.* Robotics and Autonomous Systems, vol. 55, no. 5, pp. 372–382, Mai 2007.

[Murillo 07b] A.C. Murillo, J.J. Guerrero & C. Sagüés. *SURF features for efficient robot localization with omnidirectional images.* IEEE International Conference on Robotics and Automation, ICRA'07, pp. 3901–3907, Rome, Italie, Avril 2007.

[Nayar 97] S.K. Nayar. *Catadioptric Omnidirectional Camera.* IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 482–488, Juin 1997.

[Neff 07] A.E. Neff, L. DongBin, V.K. Chitrakaran, D.M. Dawson & T.C. Burg. *Velocity control for a quad-rotor UAV fly-by-camera interface.* SoutheastCon, pp. 273–278, Richmond, USA, Mars 2007.

[Nistér 06] D. Nistér, O. Naroditsky & J. Bergen. *Visual Odometry for Ground Vehicle Applications.* Inaugural issue of Journal of Field Robotics, vol. 23, no. 1, pp. 3–20, Janvier 2006.

[Nistér 04] D. Nistér. *An efficient solution to the five-point relative pose problem.* Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 6, pp. 756–770, Juin 2004.

[Novales 94] C. Novales. *Pilotage par actions réflexes et navigation locale de robots mobiles rapides.* Thèse de doctorat, Université Montpellier II, 1994.

[Nuttin 01] M. Nuttin, E. Demeester, D. Vanhooydonck & H. Van Brussel. *Shared autonomy for wheelchair control : Attempts to assess the user's autonomy.* Autonome Mobile Systeme, pp. 127–133, 2001.

[O'Keefe 78] J. O'Keefe & L. Nadel. The hippocampus as a cognitive map. Oxford University Press, 1978.

[Pajdla 97] T. Pajdla, T. Werner & V. Hlaváč. *Correcting Radial Lens Distortion without Knowledge of 3-D Structure.* Rapport technique, Czech Technical University, Prague, République Tchéque, 1997. TR K335-CMP-1997-138.

[Pajdla 99] T. Pajdla & V. Hlaváč. *Zero phase representation of panoramic images for image based localization.* 8th International Conference on Computer Analysis of Images and Patterns, pp. 550–557, Ljubljana, Slovénie, Septembre 1999.

[Pajdla 01] T. Pajdla, T. Svoboda & V. Hlaváč. Panoramic vision : Sensors, theory and applications, chapitre Epipolar Geometry of Central Panoramic Catadioptric Cameras, pp. 73–102. R. Benosman and S. B. Kang, 2001.

[Persson 04] P.-O. Persson & G. Strang. *A Simple Mesh Generator in MATLAB.* SIAM Review, vol. 46, no. 2, pp. 329–345, Juin 2004.

[Pirjanian 99] P. Pirjanian. *Behavior Coordination Mechanisms - State-of-the-art.* Rapport technique, Robotics Research Laboratory, University of Southern California, Octobre 1999.

[Pissard-Gibollet 91] R. Pissard-Gibollet & P. Rives. *Asservissement visuel appliqué à un robot mobile : état de l'art et modélisation cinématique.* Rapport technique, INRIA, Sophia-Antipolis, Décembre 1991. RR-1577.

[Pissard-Gibollet 95] R. Pissard-Gibollet & P. Rives. *Applying visual servoing techniques to control a mobile hand-eye system.* IEEE International Conference on Robotics and Automation, ICRA'95, pp. 166–171, Nagoya, Aichi, Japon, Mai 1995.

[Pounds 02] P. Pounds, R. Mahony, P. Hynes & J. Roberts. *Design of a Four-Rotor Aerial Robot.* Australien Conference on Robotics and Automation (ACRA'02), pp. 145–150, Auckland, Nouvelle Zélande, Novembre 2002.

[Ray 94] S.F. Ray. Applied photographic optics. Focal Press, Oxford, 1994. Second edition. ISBN-0240514998.

[Redish 99] A. Redish. Beyond the cognitive map. Cambridge : MIT, 1999.

[Remazeilles 04] A. Remazeilles. *Navigation à partir d'une mémoire d'images.* Thèse de doctorat, Université de Rennes 1, 2004.

[Rivlin 03] E. Rivlin, l. Shimshonil & E. Smolyar. *Image-Based Robot Navigation in Unknown Indoor Environments.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'03, pp. 2736–2742, Las Vegas, CA, USA, Octobre 2003.

[Rottmann 05] A. Rottmann, O. Martínez Mozos, C. Stachniss & W. Burgard. *Place Classification of Indoor Environments with Mobile Robots using Boosting.* National Conference on Artificial Intelligence, Pittsburgh, PA, USA, Juillet 2005.

[Royer 07] E. Royer, M. Lhuillier, M. Dhome & J.-M. Lavest. *Monocular Vision for Mobile Robot Localization and Autonomous Navigation.* International Journal of Computer Vision, special joint issue on vision and robotics, vol. 74, pp. 237–260, Janvier 2007.

[Samson 95] C. Samson. *Control of chained systems. Application to path following and time-varying stabilization of mobile robots.* IEEE Transactions on Automatic Control, vol. 40, no. 1, pp. 64–77, Janvier 1995.

[Santos-Victor 99] J. Santos-Victor, R. Vassallo & H. Schneebeli. *Topological Maps for Visual Navigation.* First International Conference on Computer Vision Systems (ICVS'99), pp. 21–36, Las Palmas, Grandes Canaries, Espagne, Janvier 1999.

[Sarris 01] Z. Sarris. *Survey of UAV Applications in Civil Markets.* 9th Mediterranean Conference on Control And Automation, pp. 1–11, Dubrovnik, Croatie, Juin 2001.

[Scaramuzza 06] D. Scaramuzza, A. Martinelli & R. Siegwart. *A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion.* Fourth IEEE International Conference on Computer Vision Systems, ICVS 2006, New York, NY, USA, 2006.

[Silveira 08] G. Silveira, E. Malis & P. Rives. *An efficient direct approach to visual SLAM.* IEEE Transactions on Robotics (Special issue on Visual SLAM), vol. 24, no. 5, pp. 969–979, Octobre 2008.

[Simhon 98] S. Simhon & G. Dudek. *A global topological Map formed by Local Metric Maps.* International Conference on Intelligent Robots and Systems, pp. 1708–1714, Victoria, Canada, Octobre 1998.

[Smith 92] W.J. Smith. Modern lens design : A resource manual. McGraw-Hill, New-York, 1992. ISBN : 0070591784.

[Smith 99] P.W. Smith, K.B. Johnson & M.A. Abidi. *Efficient techniques for wide-angle stereo vision using surface projection models.* IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 113–118, Fort Collins, USA, Juin 1999.

[Solà i Ortega 05] J. Solà i Ortega, T. Lemaire, M. Devy, S. Lacroix & A. Monin. *Delayed vs Undelayed Landmark Initialization for Bearing Only SLAM.* ICRA'05, Workshop on Simultaneous Localization and Mapping, Barcelone, Espagne, Avril 2005.

[Stevenson 95] D.E. Stevenson & M.M. Fleck. *Nonparametric correction of distortion.* Rapport technique, Computer Science. University of Iowa. USA, 1995. TR 95–07.

[Svoboda 98] T. Svoboda, T. Pajdla & V. Hlavác. *Epipolar Geometry of Panoramic Cameras.* ECCV (1), pp. 218–231, 1998.

[Svoboda 01] T. Svoboda & T. Pajdla. *Matching in Catadioptric Images with Appropriate Windows and Outliers Removal.* 9th International Conference on Computer Analysis of Images and Patterns, pp. 733–740, Berlin, Allemagne, Septembre 2001.

[Svoboda 02] T. Svoboda & T. Pajdla. *Epipolar geometry for central catadioptric cameras.* International Journal of Computer Vision, vol. 49, no. 1, pp. 23–37, Août 2002.

[Tamimi 05] A. Tamimi, H. Andreasson, A. Treptow, T. Duckett & A. Zell. *Localization of mobile robots with omnidirectional vision using particle filter and iterative SIFT.* 2nd European Conference on Mobile Robots (ECMR), pp. 2–7, Ancona, Italie, Septembre 2005.

[Tapus 06] A. Tapus & R. Siegwart. *A Cognitive Modeling of Space using Fingerprints of Places for Mobile Robot Navigation.* IEEE International Conference on Robotics and Automation, ICRA'06, pp. 1188–1193, Orlando, USA, Mai 2006.

[Tardif 08] J.P. Tardif, Y. Pavlidis & K. Daniilidis. *Monocular Visual Odometry in Urban Environments Using an Omnidirectional Camera.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'08, pp. 2531–2538, Nice, France, Septembre 2008.

[Tessier 06] C Tessier, C. Cariou, C. Debain, F. Chausse, R. Chapuis & C. Rousset. *A Real-Time, Multi-Sensor Architecture for Fusion of Delayed Observations : Application to Vehicle Localisation.* 9th International IEEE Conference on Intelligent Transportation Systems, pp. 1316–1321, Toronto, Canada, Septembre 2006.

[Thrun 02] S. Thrun. *Robotic Mapping : A Survey.* G. Lakemeyer & B. Nebel, editeurs, Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann, 2002.

[Thuilot 04] B. Thuilot, J. Bom, F. Marmoiton & P. Martinet. *Accurate automatic guidance of an urban electric vehicle relying on a kinematic GPS sensor.* 5th IFAC Symposium on Intelligent Autonomous Vehicles, IAV'04, Instituto Superior Técnico, Lisbonne, Portugal, Juillet 2004.

[Tolman 48] E.C. Tolman. *Cognitive maps in rats and men.* The Psychological Review, vol. 55, no. 4, pp. 189–208, 1948.

[Tomatis 01] N. Tomatis, I. Nourbakhsh & R. Siegwart. *Combining Topological and Metric : A Natural Integration for Simultaneous Localization and Map Building.* Fourth European Workshop on Advanced Mobile Robots (Eurobot 2001), Lund, Suède, Septembre 2001.

[TRO 08] *Special Issue on Visual SLAM.* IEEE Transactions on Robotics, vol. 24, oct 2008.

[Tulving 85] E. Tulving. *How many memory systems are there ?* American Psychologist, vol. 40, no. 4, pp. 385–398, 1985.

[Tversky 01] N. Tversky. *Structures of mental spaces.* 3rd International Space Syntax Symposium, Atlanta, USA, 2001.

[Ulrich 00] I. Ulrich & I. Nourbakhsh. *Appearance-Based Place Recognition for Topological Localization.* IEEE International Conference on Robotics and Automation, ICRA'00, pp. 1023–1029, San Francisco, USA, Avril 2000.

[Valgren 06] C. Valgren, A. Lilienthal & T. Duckett. *Incremental Topological Mapping Using Omnidirectional Vision.* IEEE/RSJ Interna-

tional Conference on Intelligent Robots and Systems, IROS'06, pp. 3441–3447, Pékin, Chine, Octobre 2006.

[Vassallo 00] R. Vassallo, H.J. Schneebeli & J. Santos-Victor. *Visual servoing and appearance for navigation.* Robotics and Autonomous Systems, vol. 31, pp. 87–97(11), Avril 2000.

[Victorino 02] Alessandro C. Victorino. *La commande référencée capteur : une approche robuste au problème de navigation, localisation et cartographie simultanées pour un robot d'intérieur.* Thèse de doctorat, Université de Nice-Sophia Antipolis, France, 2002.

[Victorino 05] A.C. Victorino & P. Rives. *Global consistency mapping with a hybrid representation.* IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'05, pp. 2554–2559, Edmunton, Canada, Août 2005.

[Šegvić 07] S. Šegvić, A. Remazeilles, A. Diosi & F. Chaumette. *Large scale vision based navigation without an accurate global reconstruction.* IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–8, Minneapolis, USA, Juin 2007.

[Waslander 05] S. Waslander, G. Hoffmann, J.S. Jang & C. Tomlin. *Multi-Agent X4-Flyer Testbed Control Design : Integral Sliding Mode vs. Reinforcement Learning.* IEEE International Conference on Intelligent Robotics and Systems, IROS'05, pp. 468–473, Edmonton, Canada, Août 2005.

[Weng 92] J. Weng, P. Cohen & M. Herniou. *Camera calibration with distortion models and accuracy evaluation.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 14, no. 10, pp. 965–980, Octobre 1992.

[Wor 08] *Workshop on Future Directions in Visual Navigation.* IEEE International Conference on Robotics and Automation, ICRA'08, Pasadena, CA, USA, Mai 2008.

[Wu 05] A.D. Wu, E.N. Johnson & A.A. Proctor. *Vision-aided inertial navigation for flight control.* AIAA Guidance, Navigation and Control Conference and Exhibit, San Francisco, CA, USA, Août 2005.

[Xiong 97] Y. Xiong & K. Turkowski. *Creating Image-Based VR Using a Self-Calibrating Fisheye Lens.* Conference on Computer Vision and Pattern Recognition, CVPR '97, pp. 237, Puerto Rico, USA, Juin 1997.

[Yamauchi 97] B. Yamauchi & P. Langley. *Spatial Learning for Navigation in Dynamic Environments.* Journal of Robotic Systems, Special Issue on Mobile Robots, vol. 14, no. 2, pp. 107–120, Juin 1997.

[Ying 04] X. Ying & Z. Hu. *Can We Consider Central Catadioptric Cameras and Fisheye Cameras within a Unified Imaging Model.* European Conference on Computer Vision, ECCV'2004, vol. I, pp. 442–455, Prague, République Tchèque, Mai 2004.

[Zhang 98] Z. Zhang. *A Flexible New Technique for Camera Calibration.* Rapport technique, Microsoft Research, 1998. MSR-TR-98-71.

[Zhang 02] J.R. Zhang, S.J. Xu & A. Rachid. *Path tracking control of vehicles based on Lyapunov approach.* American Control Conference, vol. 3, pp. 2132–2137, Anchorage, Alaska, USA, Mai 2002.

[Zheng 92] J. Zheng & S. Tsuji. *Panoramic Representation for Route Recognition by a Mobile Robot.* International Journal of Computer Vision, no. 9, pp. 55–76, Octobre 1992.

[Zhou 03] C. Zhou, Y. Wei & T. Tan. *Mobile robot self-localization based on global visual appearance features.* IEEE International Conference on Robotics and Automation, ICRA'03, pp. 1271–1276, Taipeh, Taiwan, Septembre 2003.

[Zivkovic 06] Z. Zivkovic, B. Bakker & B. Kröse. *Hierarchical Map Building and Planning based on Graph Partitioning.* IEEE International Conference on Robotics and Automation, ICRA'06, pp. 803–809, Orlando, USA, Mai 2006.

Table des figures