Table des matières

Introduction

Si l'on effectue un saut dans le temps, et que l'on retourne en 1908, devant l'usine de Monsieur Henry Ford, nous pouvons alors assister à la sortie de la première automobile produite en quantité industrielle, la *Ford T*. Sur ce modèle, pas encore d'ouverture centralisée des portes. D'ailleurs à quoi bon puisqu'il suffit d'enjamber la portière pour rentrer dans le véhicule. Le problème du trousseau de clés toujours égaré ne se posait pas encore : pour lancer le moteur, une bonne manivelle suffit, avec un peu d'huile de coude bien sûr. Ce type de véhicule se prête particulièrement à la ballade, avec sa vitesse de croisière avoisinant les 75 km/heure.

La comparaison des voitures du siècle dernier avec celles que nous utilisons aujourd'hui montre clairement les progrès effectués dans le domaine de l'industrie automobile, fruits de travaux en mécanique, automatique, physique, électronique, ... La barrière entre la science-fiction et la réalité se déplace au gré des innovations technologiques de l'industrie automobile.

Si les bienfaits de l'automobile pour notre société sont indiscutables, l'augmentation du trafic routier engendre aussi de bien néfastes conséquences. Les accidents de la route, généralement dus à l'inattention du conducteur ou à une mauvaise infrastructure, et la saturation des réseaux routiers des grandes villes deviennent des problématiques majeures, auxquelles à la fois l'industrie automobile et le secteur public tentent d'apporter des réponses.

La solution proposée par le monde industriel est de rendre les automobiles plus sécurisées. Les avancées technologiques en matière d'électronique embarquée permettent de nos jours d'intégrer de véritables ordinateurs de bord dans les véhicules. Parmi les différentes fonctions que peut remplir cette « intelligence embarquée », citons par exemple la correction électronique de trajectoires avec le système EPS (*Electric Power Steering*), ou encore le maintien de distance de sécurité avec les autres véhicules par l'ACC (*Advanced Cruise Control*). La voiture n'est plus un simple moyen de locomotion, mais devient un véritable acteur qui possède sa part de décision durant la navigation.

De son côté, le secteur public apporte une solution claire et précise aux problèmes engendrés par l'automobile : « Utilisez moins votre voiture ! ». La mise à disposition de nombreux moyens de transport public est en effet une solution pour obtenir une diminution du trafic routier et, par la même occasion, des accidents. Pour satisfaire le besoin d'autonomie des usagers, de nombreux projets se sont intéressés à l'élaboration de systèmes de transport public individuel. Des essais grandeur nature ont été effectués en France et dans d'autres grandes villes européennes. Le projet PRAXITÈLE[1], à Saint Quentin en Yvelines en 1997, ou encore le projet TIP[2] (pour *Transport Individuel Public*) de la ville de Genève débuté en 2003 en sont des exemples. Des véhicules électriques sont mis à la disposition des usagers en centre-ville, via des systèmes d'abonnement

[1] `http://www-rocq.inria.fr/praxitele/`

[2] `http://www.ville-ge.ch/geneve/amenagement/act_transport_individuel.htm`

par carte. Là encore, des technologies de pointe sont mises en place : recharge électrique automatique des véhicules par induction, réseaux de communication centralisés, système de localisation par GPS, conduite en train avec accrochage immatériel des véhicules vides, parking automatique en créneau, ...

Et nous arrivons à la question que l'on pose très souvent aux roboticiens : « À quand la voiture sans conducteur ? ». Là encore, ce qui était de la science-fiction il y a peu se rapproche de plus en plus du domaine de la réalité. De nombreux projets de recherche présentent des résultats de déplacements autonomes de véhicules. Citons par exemple le projet américain NAVLAB[3], ou encore le projet européen PROMETHEUS[4].

Si ces travaux présentent des briques essentielles à la navigation autonome de véhicules, ils ne leur donnent pas encore une indépendance complète. Bien que la définition de stratégies de navigation soit primordiale, elle ne constitue qu'une des nombreuses aptitudes que doit posséder un système robotique pour être autonome. En effet, la complexité du problème abordé nécessite l'emploi de savoir-faire propres à différentes communautés de recherche. Preuves en sont les projets MOBIVIP[5] et BODEGA[6], qui regroupent à l'échelle nationale différents acteurs de la recherche afin de favoriser l'élaboration de stratégies globales de navigation autonome.

Un système de navigation autonome doit pouvoir répondre aux trois mêmes questions que nous nous posons tous inconsciemment lorsque nous utilisons nos véhicules :

- Où suis-je ?
- Où dois-je aller ?
- Comment m'y rendre ?

Les deux premières questions posent le problème de la localisation. Cette localisation nécessite le choix de capteurs afin que le robot puisse reconnaître son environnement alentour. Et qui dit reconnaissance dit fatalement présence d'une connaissance initiale. Tout comme le conducteur humain utilise sa mémoire pour se localiser, un système robotique autonome doit être pourvu d'une représentation interne de son environnement de navigation.

La troisième question impose une contrainte additionnelle sur la représentation interne de l'environnement. Si la mémoire du robot doit lui permettre de se localiser, elle doit aussi lui donner la capacité de se définir un plan de navigation pour atteindre son objectif. Cette aptitude passe par la mise en place d'une organisation adéquate de la mémoire du robot. Elle doit lui permettre de faire le lien entre sa position initiale et la position qu'il désire atteindre.

Les réponses à ces trois questions suffisent aux conducteurs humains pour réaliser une tâche de navigation. Tout conducteur est censé posséder son permis de conduire ; le contrôle des mouvements du véhicule n'est donc pas, *a priori*, une difficulté. Cependant, donner aux robots la capacité de contrôler leurs mouvements reste un problème majeur de la recherche en robotique. Une quatrième question, de taille, doit donc être rajoutée pour pouvoir réaliser une tâche de navigation par un système robotique : « Comment réaliser ce déplacement ? ».

[3] `http://www.navlab.org/`

[4] `http://cmm.ensmp.fr/~beucher/prometheus.html`

[5] `http://www-sop.inria.fr/mobivip/` : ce projet s'inscrit dans le cadre du programme PREDIT de recherche, d'expérimentation et d'innovation dans les transports terrestres, initié et conduit par les ministères chargés de la recherche, des transports, de l'environnement et de l'industrie, l'ADEME et l'ANVAR.

[6] `http://jazz.ensil.unilim.fr/bodega/` : projet s'inscrivant dans le cadre de ROBEA (*ROBotique et Entités Artificielles*), programme interdisciplinaire de recherche initié par le CNRS et soutenu par l'INRIA.

De cette analyse, nous en déduisons qu'un système robotique autonome doit pour réaliser une tâche de navigation :

- posséder un (ou des) capteur(s) extéroceptif(s) ;
- avoir une représentation de son environnement de navigation ;
- être capable de se localiser dans cet environnement ;
- posséder un outil de définition d'un plan de navigation ;
- être capable de contrôler ses mouvements pour réaliser ce plan.

Le sujet de cette thèse s'inscrit donc dans la définition d'un formalisme de navigation robotique autonome ; elle se doit donc de satisfaire ces différents pré-requis. Afin de justifier les choix effectués dans notre approche, le reste de cette introduction se focalise sur les différentes réflexions ayant amené au formalisme proposé.

Les capteurs pour la navigation

Parmi les différents capteurs existant sur le marché, les systèmes GPS (*Global Positionning System*) apparaissent naturellement comme des capteurs très intéressants [Cordesses 00, Katsura 03] ; ils permettent en effet d'effectuer une localisation géo-référencée du capteur, par un processus de triangulation d'ondes émises par des satellites. Combinés avec une carte du réseau routier, ils peuvent réaliser une localisation précise d'un véhicule [ElNajjar 03].

Néanmoins, de nombreuses conditions doivent être satisfaites pour que ces systèmes puissent être utilisés efficacement. Ainsi, leur bon fonctionnement passe non seulement par une bonne couverture satellitaire, mais aussi par une topologie adéquate de l'environnement. En effet, ce système reste encore aujourd'hui inopérant si le capteur se situe dans une zone où il ne « voit » pas directement les satellites qui lui permettent de se localiser. La mise en service du système GALILEO [7] permettra d'avoir dans un futur proche plus de satellites à disposition ; de nouveaux protocoles de système GPS s'intéressent au dur problème de la réflexion des ondes sur les bâtiments ; certains travaux scientifiques considèrent les problèmes de localisation en environnement urbain et encombré [Cui 03]. Si l'avenir de ce système de localisation semble prometteur, il reste cependant aujourd'hui difficile d'effectuer une localisation correcte et précise si le capteur est situé dans une zone encaissée, comme dans l'exemple typique du « canyon urbain », où la route est encadrée par de hauts bâtiments.

Les capteurs de type laser permettent aisément d'obtenir une carte de profondeur d'un environnement. Le laser est très souvent employé pour les opérations de localisation et de modélisation simultanées dans des environnements intérieurs [Thrun 00, Guivant 01, Victorino 01, Victorino 04]. Il peut aussi être utilisé pour effectuer des reconstructions denses et très précises d'objets ou de bâtiments [Curless 96]. Les systèmes de localisation utilisant un tel capteur sont basés sur l'hypothèse que la structure géométrique de l'environnement dans lequel se situe le système robotique est une information suffisante pour localiser le robot. Or la majeure partie des rues d'une ville possède la même topologie, à savoir une route entourée de bâtiments. Dans ces cas de figure, le laser peut connaître des difficultés pour distinguer une rue d'une autre de manière efficace. Les ambiguïtés sont généralement levées en déplaçant le système robotique, afin que l'évolution de la structure géométrique de la scène durant le mouvement lui permette de mieux caractériser sa position. Cependant, la localisation peut s'avérer délicate si la structure de

[7] http ://europa.eu.int/comm/dgs/energy_transport/galileo/ : système de radionavigation par satellite, lancée par l'Union Européenne et l'Agence Spatiale Européenne (ESA).

l'environnement proche n'est pas assez discriminante pour le différencier du reste de la scène, et si cette localisation doit s'effectuer sans mouvement et sans utilisation d'information *a priori* sur la position courante du système robotique.

Les capteurs de vision sont de plus en plus employés pour réaliser des tâches de déplacement autonome de systèmes robotiques [Chaumette 90, Hutchinson 96, Malis 00, Burschka 01, Blanc 04]. La richesse de l'information contenue dans les images rend les caméras adaptées à une très grande variété d'environnements. De plus, de nombreux systèmes de localisation basés vision ont été proposés et démontrent ainsi l'efficacité de ce capteur [Ulrich 00, Se 02, Wolf 02, Košecká 03]. Les caméras catadioptriques, encore appelées caméras omnidirectionnelles, apparaissent particulièrement performantes pour les opérations de navigation de systèmes robotiques [Gaspar 00, Matsumoto 00a, Gaspar 00, Paletta 01, Menegatti 04, Mezouar 04]. L'un des principaux avantages de ces capteurs est qu'ils fournissent une description de 360 degrés de l'environnement. Cette vue panoramique permet de minimiser l'impact de la présence d'éventuels obstacles inconnus, puisqu'ils n'occupent qu'une petite zone dans les images. De nombreux robots mobiles d'intérieur utilisent ce genre de capteurs pour se localiser ou bien se déplacer, la caméra étant généralement située sur le dessus du robot. Si une telle installation est facilement réalisable pour cette classe de robots, elle apparaît en revanche beaucoup plus difficile à mettre en œuvre pour un véhicule de la taille d'une automobile. La seule manière de conserver le champ de vision de ces caméras serait en effet de les percher au dessus du toit de la voiture. Dans une telle position, le capteur ne permettrait alors pas de décrire l'environnement qui se situe au niveau du sol, comme les piétons par exemple. De plus, ce type de capteur reste encore aujourd'hui beaucoup plus fragile et onéreux qu'une caméra classique.

Une description de l'environnement valide pour la localisation et la navigation

Les problèmes de localisation et de navigation ont pendant de très longues années été considérés de manière indépendante par la communauté scientifique. La complexité de chacune de ces deux tâches nécessitait, à raison, de se concentrer sur l'une ou l'autre problématique, avant de pouvoir espérer les enchaîner.

Bien que traitées indépendamment, ces deux techniques imposaient un même pré-requis, à savoir la connaissance du modèle $3D$ de l'environnement. Soit ce modèle est connu *a priori* [Koditschek 87, Thrun 96, Arikan 01, Yuen 02, Dao 03], soit il est reconstruit durant une phase autonome de reconstruction [Victorino 00, Burschka 01, Se 02, Davison 03, Royer 04]. La reconstruction $3D$ n'est cependant pas une opération triviale et reste encore un axe de recherche scientifique très actif. Bien que les progrès scientifiques, tant sur le formalisme théorique de reconstruction que sur la puissance des ordinateurs, portent à espérer que cette tâche puisse bientôt être traitée « à la volée », la reconstruction $3D$ n'en reste pas moins un problème ouvert.

L'une des difficultés auxquelles doivent se confronter les techniques basées modèle (reconstruction, localisation ou navigation) vient de la nécessité de traduire l'information initialement obtenue dans l'espace de mesure du capteur vers un repère global tridimensionnel lié à la scène. Pourtant, cette traduction n'est pas impérative pour effectuer des tâches de localisation ou de navigation. Preuve en sont les nombreux travaux opérant directement dans l'espace des mesures des capteurs. Les travaux de reconnaissance d'images permettent, pour une image « inconnue », de détecter automatiquement dans une base d'images celles qui lui sont visuellement proches [Ulrich 00, GonzalesBarbosa 02, Wolf 02, Košecká 03, Zhou 03]. Une localisation est donc réali-

sable directement à partir de prises de vue sans nécessiter une reconstruction de l'environnement.

Les approches de navigation par asservissement visuel montrent qu'un schéma de contrôle des mouvements d'un système robotique peut être directement établi à partir de données images [Malis 99, Mezouar 02b, Tahri 04]. Le modèle de la scène n'est donc pas un prérequis impératif.

Ces caractéristiques ne sont pas nouvelles, et ont déjà été mises à profit dans de nombreuses applications. Cependant, rares sont les travaux permettant de réaliser conjointement la localisation et la navigation dans un même formalisme. La représentation de l'environnement est donc fortement orientée localisation ou navigation, mais rarement les deux. La plupart des techniques de localisation basée apparence représentent l'environnement par un ensemble de prises de vue [Matsumoto 00b, Wolf 02, Košecká 03, Menegatti 04]. L'organisation de cette base consiste généralement à représenter les images dans un espace associé aux descripteurs images utilisés. Si cette représentation est performante pour la localisation, elle ne permet pas de définir le chemin que doit réaliser le système robotique pour atteindre une position. Les techniques de navigation robotique basée vision représentent un environnement par un ensemble de prises de vue à des positions clés de l'environnement. Si ces images permettent de suivre et contrôler l'avancement du système robotique, elles ne lui donnent généralement pas la possibilité de se localiser de manière autonome avant le début de son déplacement.

Localisation quantitative et localisation qualitative

La localisation quantitative et la localisation qualitative se différencient par la précision recherchée. La première consiste à localiser de manière fine le robot, généralement en estimant la pose du robot par rapport à un repère lié au modèle de la scène [Cobzas 01, Wolf 02, Dao 03]. La localisation qualitative effectue « simplement » une mise en relation des informations du capteur avec celles stockées dans la base. Par exemple, ces techniques fournissent les images de la base qui, en terme de contenu visuel, sont proches de la prise de vue fournie par la caméra [Kröse 99, Sim 99, Paletta 01, Menegatti 04]. Puisque les informations visuelles contenues dans une image dépendent de la position de la caméra qui l'a acquise, déterminer les images les plus proches de la base permet d'avoir une information qualitative de la position du robot. Comme nous l'avons déjà signalé, les techniques d'asservissement visuel permettent de contrôler les mouvements d'un système robotique à partir d'images, même si la pose du robot est inconnue [Chaumette 90, Hutchinson 96]. En principe, il n'est donc pas nécessaire de connaître la position précise du robot pour effectuer une tâche de navigation.

Localisation AVANT le mouvement et PENDANT le mouvement

Bien que ces deux localisations nous semblent bien différentes, elles ne sont généralement pas clairement dissociées dans la littérature. Un système robotique doit pouvoir se localiser avant de commencer son déplacement afin justement de définir son plan de navigation. Dans la plupart des techniques de navigation basées vision, cette localisation initiale n'est que très rarement effectuée : la position initiale du robot est en effet souvent supposée connue [Matsumoto 00a, Argyros 01, Cobzas 03, Smith 02]. Naturellement, cette hypothèse est convenable dans de nombreux cas de figure. Cependant, si l'on donne aux systèmes robotiques la capacité d'effectuer cette localisation initiale, ceux-ci ne seront que plus autonomes.

D'un autre coté, de nombreuses méthodes de localisation s'aident du mouvement du robot afin de définir précisément sa position dans l'espace [Burgard 98, Victorino 00, Thrun 00]. La

comparaison du modèle global de l'environnement avec le modèle local, déduit des informations du ou des capteurs, ne donne généralement pas qu'une seule position possible pour le robot, et son déplacement permet de lever ces ambiguïtés. Finalement, ces techniques permettent de localiser le robot pendant son déplacement et non pas avant celui-ci.

Les techniques proposant de localiser un robot sans qu'il ait à se déplacer sont généralement basées sur une description fine des images fournies par la caméra [Kröse 99, Sim 99, Paletta 01, Menegatti 04]. Cependant, plus la taille de la scène est grande, plus ces méthodes deviennent coûteuses en temps de calcul. Elles semblent donc difficilement utilisables pour la localisation du système robotique durant sa navigation, où le temps de réponse est une contrainte forte.

La localisation initiale du système robotique doit permettre de créer un lien avec la position qu'il doit atteindre. Par ce lien, il est alors possible de délimiter l'environnement local que doit traverser le robot pour atteindre son but. Si le système robotique doit pouvoir se localiser avant son déplacement dans l'ensemble de la scène, durant le déplacement il n'a besoin de connaître son avancement que par rapport au chemin qu'il s'est donné pour atteindre son but.

Schéma de navigation contraint ou souple

Les problèmes de navigation ont souvent été décomposés en deux sous-problèmes : la planification de trajectoire, puis le suivi de cette trajectoire [Laumond 01, Siegwart 04]. De manière générale, la planification de trajectoire pour la navigation impose la connaissance du modèle $3D$ de l'environnement. Cette contrainte forte limite les contextes d'application et d'utilisation de ces formalismes.

Contrairement aux tâches de positionnement, une tâche de navigation est définie par des positions initiale et désirée complètement différentes. En ce sens, les informations fournies par les capteurs extéroceptifs ne peuvent être mises directement en relation.

Deux grandes familles d'approches permettent de résoudre ce problème de discontinuité de l'information capteur. L'une d'entre elles consiste à définir les positions intermédiaires que doit successivement atteindre le système robotique [Rasmussen 96, Argyros 01, Blanc 04]. Un tel *modus operandi* nécessite de posséder une base d'information décrivant l'environnement de manière très précise, afin que soient fournies au système robotique les meilleures positions intermédiaires. La seconde approche proposée revient à sauvegarder la trajectoire des primitives visuelles durant la phase d'apprentissage [Gaspar 00, Burschka 01, Mezouar 02c]. Le robot est alors contraint à répéter ces trajectoires durant son déplacement.

Une définition précise de la trajectoire à suivre rend difficile l'ajout de contraintes additionnelles liées par exemple à la propriété dynamique des environnements urbains. Au contraire, la définition d'un schéma de navigation plus souple permet plus aisément de rajouter des contraintes durant la navigation. De la même manière, l'utilisation d'un schéma de contrôle en ligne assure plus de souplesse au système. Si l'on veut rajouter des propriétés particulières au système robotique, il n'est pas nécessaire de retraiter toute la base d'information pour obtenir des trajectoires ou des positions clés adaptées. Seule la méthode de contrôle de la navigation a alors besoin d'être mise à jour.

Approche proposée

Le formalisme que nous proposons s'inscrit dans la grande famille des techniques basées apparence. Dans notre approche, nous ne supposons pas connaître le modèle tridimensionnel de la scène, et nous ne cherchons pas non plus à l'obtenir. La représentation de l'environnement correspond à une collection d'images acquises par la caméra embarquée du système robotique durant une phase d'apprentissage où le robot est contrôlé par un opérateur. Cette base d'information est ensuite organisée de manière automatique. La représentation interne permet d'effectuer une localisation qualitative du système robotique avant son déplacement (voir la figure 0.1). Elle nous permet aussi de définir un chemin d'images qui met visuellement en relation l'image fournie par la caméra avant le déplacement avec celle qu'elle doit fournir à la fin de son mouvement. Le chemin d'images constitue donc pour le système robotique une description dense de l'environnement visuel que doit observer la caméra durant la navigation (figure 0.2).

La méthode de contrôle des mouvements du robot que nous proposons ne nécessite pas de planification de trajectoire hors ligne. Les déplacements sont calculés en ligne, en fonction de l'image acquise par le système robotique et du chemin d'images (figure 0.3). Aucune autre information n'est donnée au système. Le robot ne cherche pas à atteindre les positions intermédiaires associées à ces images. Ces dernières décrivent les primitives (encore appelées amers) de l'environnement qui doivent être successivement observées par la caméra durant le déplacement. Le robot se déplace donc afin de rendre observables les amers qui ne le sont pas actuellement dans l'image acquise par la caméra.

Durant la navigation, nous ne cherchons pas à localiser le robot par rapport à la base d'images de l'environnement (notons que cette localisation pourrait être effectuée dans une boucle asynchrone, comme dans [Parra 99]). Grâce au chemin d'images, celui-ci possède en effet toutes les informations nécessaires pour se déplacer. Dans notre contexte, la localisation durant la navigation consiste à mettre en relation les amers visuels contenus dans le chemin d'images avec ceux suivis dans la prise de vue fournie par la caméra. Cette mise en relation nous permet de détecter si de nouveaux amers sont rentrés dans le champ de vision de la caméra, afin de les prendre en compte dans le schéma de suivi et dans le schéma de navigation.

Contributions scientifiques

Notre objectif n'était pas de fournir un système de navigation autonome prêt à l'emploi. De la même manière, nous ne prétendons pas que l'utilisation d'une caméra permettra de résoudre tous les problèmes de la navigation robotique autonome.

Cependant, nous estimons que le capteur caméra n'a pas encore dévoilé toutes ses capacités. En plaçant ce capteur au centre de toutes les opérations permettant de définir et réaliser une tâche de navigation, les travaux réalisés durant cette thèse proposent de nouvelles manières d'exploiter les informations fournies par une caméra.

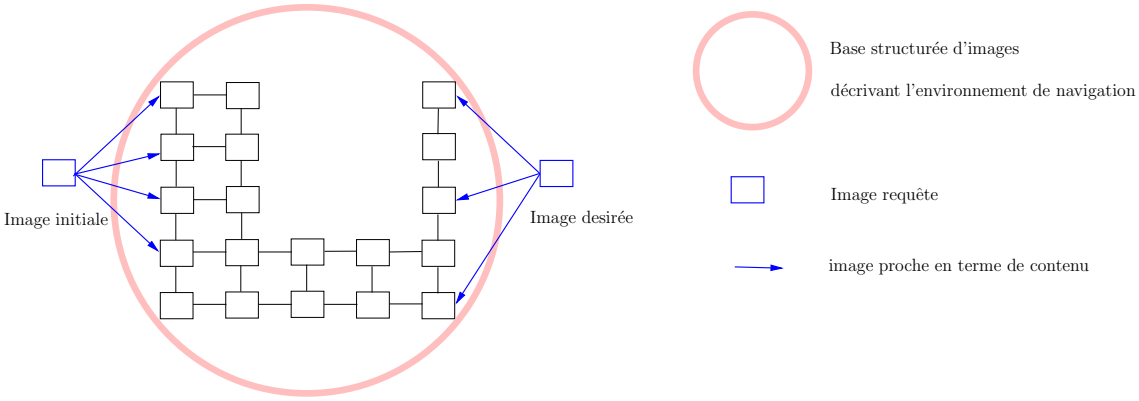


FIG. 0.1: Approche proposée, questions « *Où suis-je ?* » et « *Où dois-je aller ?* » : la comparaison des images initiale et désirée avec la base d'images décrivant l'environnement de navigation permet de déterminer les vue les plus semblables en terme de contenu.

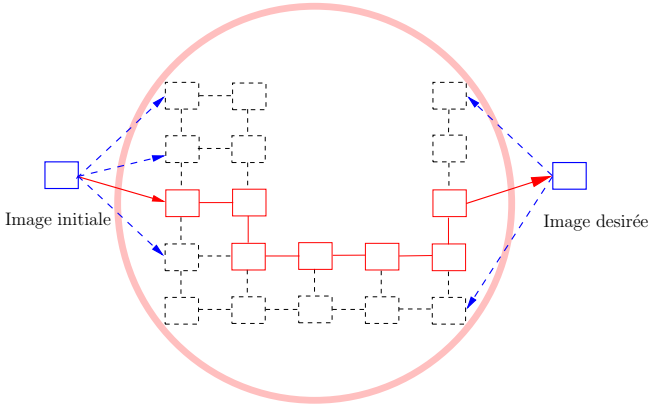


FIG. 0.2: Approche proposée, question « *Comment m'y rendre ?* » : recherche dans la base structurée d'un chemin d'images décrivant de manière continue l'espace séparant les positions initiale et désirée.

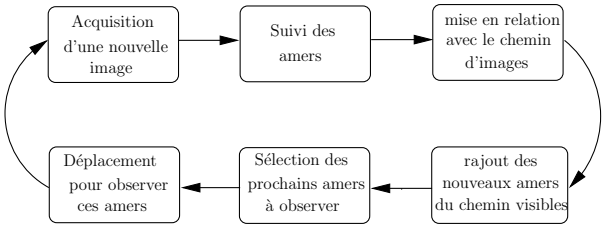


FIG. 0.3: Approche proposée, question « *Comment naviguer dans ce chemin ?* » : une boucle fermée réalisée en ligne permet de définir pour chaque image acquise par la caméra embarquée le meilleur mouvement à réaliser pour se rapprocher de la position désirée

De ce fait, la contribution majeure de cette thèse est la définition d'un formalisme homogène basé vision permettant de traiter le problème de la navigation depuis la représentation de l'environnement jusqu'au contrôle en ligne des mouvements du système robotique. Plus précisément, ces travaux de thèse concernent :

- la définition d'un schéma d'organisation d'une base d'images pour la localisation et la navigation. Nous proposons de plus un formalisme permettant de simplifier la recherche de chemin dans la base, tout en prenant en compte les contraintes de mouvement des robots mobiles ;
- la mise à jour automatique des informations visuelles durant la navigation, dans le cas d'environnements plans et tridimensionnels ;
- l'élaboration d'une méthode générique de contrôle en ligne des mouvements d'un système robotique. Elle permet de gérer de longs déplacements en n'utilisant que des informations extraites des images.

Organisation du manuscrit

La première partie de ce manuscrit effectue un rappel des fondements théoriques de la vision par ordinateur nécessaires à la compréhension des travaux présentés. Cette partie s'intéresse tout d'abord aux relations géométriques associées à une ou deux prise(s) de vue d'une même scène. Plus exactement, les notions de base sur la géométrie projective, la modélisation des caméras, ainsi que sur les représentations matricielles de la géométrie épipolaire y sont rappelées. Les principales méthodes permettant d'estimer la géométrie épipolaire, le déplacement partiel de la caméra et la structure de la scène observée sont également présentées.

Dans un second temps, cette partie s'intéresse aux traitements bas-niveaux permettant d'extraire des informations d'une prise de vue. Nous effectuons tout d'abord un rappel sur la technique de détection de points d'intérêt la plus employée, que nous utiliserons par la suite. Les principes de la mise en correspondance de points d'intérêt entre deux prises de vue d'une même scène sont ensuite introduits. Enfin nous présentons la technique de suivi de primitives qui sera utilisée pour localiser les points d'intérêt dans une nouvelle image, suite au mouvement de la caméra.

La deuxième partie de ce rapport contient les réponses aux questions « *Où suis-je ?* », « *Où dois-je aller ?* » et « *Comment m'y rendre ?* ». L'état de l'art présenté regroupe les approches utilisées dans la littérature en fonction de la représentation interne de l'environnement du robot. Nous présentons ensuite la représentation topologique d'une scène que nous proposons. L'espace de navigation est directement décrit dans l'espace image, par une base de prises de vue. Le problème de localisation induit par les deux questions précédentes est résolu de manière topologique. La position exacte du système robotique dans l'environnement n'est pas recherchée. La localisation que nous effectuons consiste à retrouver dans la base les images dont le contenu visuel est le plus semblable à celui de l'image fournie par la caméra avant le mouvement. Nous montrons que cette étape de localisation peut être traitée à l'aide de techniques ayant déjà fait leurs preuves en indexation d'images. Des résultats expérimentaux sur différentes bases d'images sont présentés.

Nous présentons ensuite le formalisme proposé pour répondre à la troisième question. À l'instar des techniques de localisation topologique, la base d'images est représentée par un graphe d'adjacence, et les chemins dans l'environnement par un succession de prises de vue extraites de la base. Nous montrons tout d'abord qu'il est possible de définir ce graphe directement à partir des primitives en correspondance entre les différentes images de la base. Une valuation permettant de

prendre en compte les contraintes de mouvements associés aux robots non-holonomes (telle une voiture) est introduite. Enfin, une méthode issue des techniques de partitionnement de graphes est adaptée à notre problématique. La recherche d'un chemin dans la base est ainsi facilitée en diminuant la taille du graphe à considérer. La validité de la valuation proposée et de la technique de recherche de chemin est prouvée par des résultats expérimentaux obtenus sur des bases d'environnements réels et simulés.

La quatrième partie aborde le problème de la mise à jour des primitives visibles durant la navigation. Dans un premier temps, un bref état de l'art montre comment l'évolution des primitives visibles est généralement abordée dans les systèmes de navigation basés vision. Nous proposons ensuite un formalisme de mise à jour utilisant des techniques de transfert d'images à partir de matrices d'homographie. Des environnements de navigation plan et tridimensionnels sont considérés. Nous montrons ensuite le schéma général de mise à jour des primitives visibles, à partir du chemin d'images décrivant la scène. Enfin des résultats expérimentaux démontrent la validité de cette approche.

La cinquième partie propose une réponse à la question « *Comment naviguer dans ce chemin ?* ». Les techniques employées dans la littérature sont tout d'abord rappelées. Nous présentons ensuite l'approche proposée. Celle-ci peut être considérée comme un *asservissement visuel qualitatif*. Elle s'inspire de la méthode des champs de potentiel, dont nous rappelons les fondements. Nous définissons ensuite les trois mesures visuelles utilisées afin de déterminer la force à appliquer au système robotique. La force engendrée permet de déplacer le système robotique pour rendre observables de nouvelles primitives connues grâce au chemin d'images. Le changement d'ensemble d'amers d'intérêt permet alors de diriger le robot vers sa position désirée. Cette partie se conclut par un ensemble de résultats expérimentaux. Les premiers ont été obtenus sur un système robotique réel. Les expériences suivantes ont été réalisées sur un simulateur, afin de se concentrer sur l'évaluation de la partie contrôle du formalisme proposé.

Chapitre 1

Fondements théoriques

Ce chapitre rappelle les bases mathématiques de la vision par ordinateur sur lesquelles s'appuient nos travaux. Il y a deux manières d'appréhender des images acquises par une caméra. La première consiste à se focaliser sur la structure de la scène observée. C'est donc la géométrie entre les primitives considérées qui motive alors les travaux. La seconde s'intéresse aux traitements bas-niveaux permettant d'extraire de ces images une information exploitable. Ces deux approches sont plus que complémentaires, puisque la première utilise comme données d'entrée les résultats obtenus par la seconde.

La première section de ce chapitre s'intéresse à l'approche géométrique de la vision par ordinateur, à savoir les relations reliant un monde tri-dimensionnel avec les informations $2D$ fournies par une ou plusieurs caméras. La deuxième partie concerne le traitement d'images.

1.1 Géométrie des caméras

La section suivante ne constitue qu'un aperçu des grandeurs géométriques définissant une ou plusieurs caméras. Le lecteur intéressé trouvera de plus amples détails dans les ouvrages [Faugeras 93, Horaud 93, Mohr 96, Hartley 00].

Cette partie commence par montrer comment les espaces géométriques projectif et euclidien peuvent être mis en relation. La modélisation des caméras est ensuite introduite, ainsi que les relations liant plusieurs images d'une même scène.

1.1.1 Relation entre l'espace euclidien et l'espace projectif

Nous supposons que le lecteur possède des notions de géométrie projective. Les ouvrages cités précédemment possèdent de riches descriptions des propriétés de cette géométrie. Nous rappelons succinctement dans cette section la relation qui peut être établie entre les espaces projectif et euclidien, tout en introduisant par la même occasion les notations qui seront utilisées dans ce manuscrit.

L'utilisation de la géométrie projective en vision par ordinateur est particulièrement intéressante puisqu'elle permet non seulement de rendre homogènes et linéaires les opérations de transformation de l'espace euclidien (dont la projection centrale, qui est le modèle le plus courant de caméra), mais aussi de travailler avec des entités géométriques à l'infini.

L'espace euclidien peut en effet être vu comme une restriction de l'espace projectif. Considérons un point $\mathcal{X}$ de l'espace euclidien $\mathbb{R}^n$ de dimension n. Il est défini par un vecteur colonne $\overline{\mathbf{X}} = (X_1, \ldots, X_n)$, exprimé dans un repère rattaché à la scène. Ce point peut être mis en relation dans l'espace projectif $\mathbb{P}^n$ avec le point $\mathbf{X} = (\lambda X_1, \ldots, \lambda X_n, \lambda)$, où $\mathbf{X}$ est de dimension $n+1$ et λ un scalaire non nul.

Les coordonnées cartésiennes d'un point de l'espace euclidien sont obtenues à partir de ses coordonnées homogènes $\mathbf{X}$ dans $\mathbb{P}^3$ en divisant toutes les coordonnées projectives par λ, si ce scalaire est non-nul. Si $\lambda = 0$, alors $\mathcal{X}$ représente une direction dans l'espace euclidien, ou un point à l'infini.

L'intérêt de l'utilisation de la géométrie projective est flagrant dès lors que l'on veut appliquer des transformations dans l'espace euclidien. Soient $\mathcal{F}_x$ et $\mathcal{F}_y$ deux repères de l'espace euclidien. Les coordonnées d'un point $\mathcal{X}$ de l'espace, respectivement ${}^x\overline{\mathbf{X}}$ et ${}^y\overline{\mathbf{X}}$ dans chacun des deux repères, satisfont la relation suivante :

$$ {}^x\overline{\mathbf{X}} = {}^x\mathbf{R}_y \, {}^y\overline{\mathbf{X}} + {}^x\mathbf{t}_y, \tag{1.1} $$

où ${}^x\mathbf{R}_y$ et ${}^x\mathbf{t}_y$ sont respectivement la matrice de rotation et le vecteur de translation permettant de passer du repère $\mathcal{F}_y$ au repère $\mathcal{F}_x$. Dans l'espace projectif, (1.1) s'écrit sous la forme d'une transformation projective ${}^x\mathbf{T}_y$ telle que :

$$ {}^x\mathbf{X} \propto {}^x\mathbf{T}_y \, {}^y\mathbf{X}, $$

où $\propto$ désigne l'égalité à un facteur d'échelle près, et :

$$ {}^x\mathbf{T}_y = \begin{bmatrix} {}^x\mathbf{R}_y & {}^x\mathbf{t}_y \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} $$

La représentation projective de cette transformation permet de rendre homogène l'équation (1.1). Le groupe des transformations de la forme de ${}^x\mathbf{T}_y$ est un sous-ensemble des transformations projectives de $\mathbb{P}^3$ vers $\mathbb{P}^3$ (de manière générale, une transformation de $\mathbb{P}^n$ vers $\mathbb{P}^n$ est appelée *homographie*). Une matrice d'homographie étant inversible, la transformation inverse ou *duale* est alors :

$$ {}^y\mathbf{T}_x = {}^x\mathbf{T}_y{}^{-1} = \begin{bmatrix} {}^x\mathbf{R}_y^\top & -{}^x\mathbf{R}_y^\top {}^x\mathbf{t}_y \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} {}^y\mathbf{R}_x & {}^y\mathbf{t}_x \\ \mathbf{0} & 1 \end{bmatrix} $$

L'utilisation de la géométrie projective est tout aussi intéressante pour exprimer des relations dans le plan. Considérons par exemple deux points x_1 et x_2 de $\mathbb{P}^2$ de coordonnées $\mathbf{x}_1$ et $\mathbf{x}_2$. La droite l passant par ces deux points est définie par le produit vectoriel suivant :

$$ \mathbf{l} \propto \mathbf{x_1} \wedge \mathbf{x_2}, $$

où $\mathbf{l}$ est le vecteur de coordonnées de la droite l. Tout point x appartenant à l vérifie alors :

$$ \mathbf{x}^\top \mathbf{l} = 0 $$

La section suivante montre comment la géométrie projective permet de la même manière de simplifier les notations de projection perspective réalisée par une caméra, sous la forme d'une transformation de $\mathbb{P}^3$ vers $\mathbb{P}^2$.

Notons que par souci de simplification des notations, nous n'effectuerons plus la différence entre les coordonnées cartésiennes $\overline{\mathbf{X}}$ (respectivement $\overline{\mathbf{x}}$) et les coordonnées homogènes $\mathbf{X}$ (respectivement $\mathbf{x}$) d'un point de l'espace (resp. du plan). Nous utiliserons la représentation homogène dans les deux cas, et indiquerons en cas d'ambiguïté l'espace considéré.

1.1.2 Modèle sténopé d'une caméra

Il existe différents modèles permettant de représenter géométriquement une caméra. Des soucis de simplification du modèle complet [Dementhon 95], et/ou les connaissances sur la structure interne de la caméra permettent de choisir le modèle adéquat. Le modèle sténopé, illustré sur la figure 1.1, est le plus couramment utilisé.

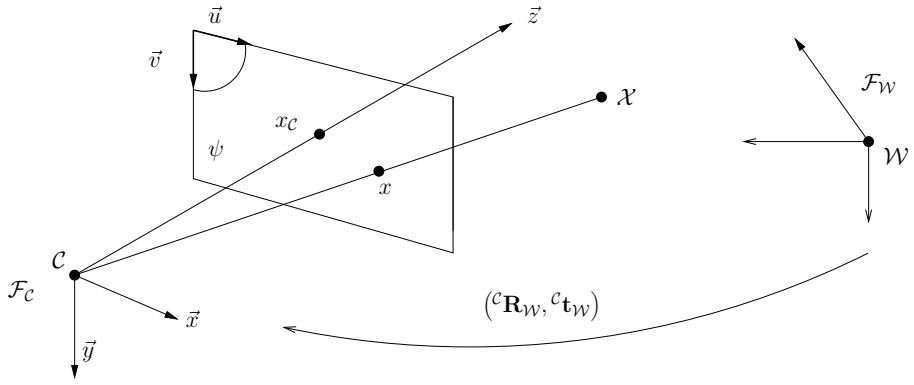


FIG. 1.1: Modèle sténopé d'une caméra

À la caméra est associé un repère orthonormé $\mathcal{F}_\mathcal{C} = (\mathcal{C}, \vec{x}, \vec{y}, \vec{z})$, dont l'origine correspond au centre de projection. Le plan image ψ est parallèle au plan $(\vec{x}, \vec{y})$, et situé à une distance f (longueur focale) de l'origine. La projection orthogonale de $\mathcal{C}$ sur ψ est le *point principal* $x_\mathcal{C}$.

Un point $\mathcal{X}$ de l'espace est défini par un vecteur de coordonnées relatif à un repère orthonormé $\mathcal{F}_\mathcal{W}$ de l'espace. On note ${}^\mathcal{W}\mathbf{X} = ({}^\mathcal{W}X, {}^\mathcal{W}Y, {}^\mathcal{W}Z, {}^\mathcal{W}T)$ ses coordonnées homogènes. La position du point dans l'image est représentée par ses coordonnées exprimées en pixel (ou *pixelliques*) notées $\mathbf{x}_p = (x, y, 1)$. Elle résultent de la succession des trois opérations suivantes :

- un changement de repère de $\mathcal{F}_\mathcal{W}$ vers $\mathcal{F}_\mathcal{C}$;
- une projection sur le plan image ;
- une transformation dans ce plan.

Le changement de repère entre $\mathcal{F}_\mathcal{W}$ et $\mathcal{F}_\mathcal{C}$ est défini par un mouvement rigide $({}^\mathcal{C}\mathbf{R}_\mathcal{W}, {}^\mathcal{C}\mathbf{t}_\mathcal{W})$. Les coordonnées ${}^\mathcal{C}\mathbf{X} = ({}^\mathcal{C}X, {}^\mathcal{C}Y, {}^\mathcal{C}Z, {}^\mathcal{C}T)$ du point dans le repère de la caméra sont obtenues par la transformation ${}^\mathcal{C}\mathbf{T}_\mathcal{W}$ associée :

$$ {}^\mathcal{C}\mathbf{X} = {}^\mathcal{C}\mathbf{T}_\mathcal{W} {}^\mathcal{W}\mathbf{X} \tag{1.2} $$

La projection $\mathbf{x}_n = (x, y, 1)$ de ce point[1] $3D$ exprimée dans le repère de la caméra sur le plan image $2D$ est déduite d'une application directe du théorème de Thalès :

$$ x = f\frac{{}^\mathcal{C}X}{{}^\mathcal{C}Z} \qquad y = f\frac{{}^\mathcal{C}Y}{{}^\mathcal{C}Z} \tag{1.3} $$

Ces relations peuvent s'écrire sous la forme d'une transformation projective $\mathbf{P_0}$ de $\mathbb{P}^3$ vers $\mathbb{P}^2$:

$$ \mathbf{P_0} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \tag{1.4} $$

[1] $\mathbf{x}_n$ désigne les coordonnées normalisées du point, exprimé dans le repère induit par $\mathcal{F}_\mathcal{C}$ sur ψ

Dans les images réelles, l'origine des coordonnées images n'est pas nécessairement le point principal et le facteur d'échelle sur chaque axe peut être différent (voir la figure 1.2). La matrice triangulaire supérieure $\mathbf{K}$ permet de passer des coordonnées métriques aux coordonnées pixelliques :

$$\mathbf{K} = \begin{bmatrix} k_x & -k_x \cot\theta & x_{p_c} \\ 0 & k_y \sin\theta & y_{p_c} \\ 0 & 0 & 1 \end{bmatrix}, \tag{1.5}$$

où :

- k_x et k_y sont les dimensions d'un pixel ;
- x_{p_c} et y_{p_c} les coordonnées pixelliques du point principal ;
- θ l'angle entre les axes $\vec{u}$ et $\vec{v}$ du repère image (généralement, on considère cet angle égal à $\pi/2$).

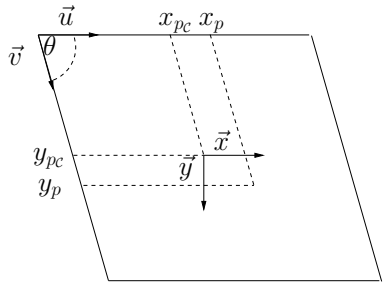


FIG. 1.2: Transformation mètre → pixel

Finalement, la transformation perspective réalisée par une caméra peut s'écrire tout simplement comme une composition des transformations projectives présentées ci-dessus :

$$\mathbf{x_p} = \mathbf{P}^{\mathcal{W}}\mathbf{X} \text{ avec } \mathbf{P} = \mathbf{K}\mathbf{P_0}{}^{\mathcal{C}}\mathbf{T}_{\mathcal{W}} \tag{1.6}$$

Notons que généralement la focale f apparaissant dans l'équation (1.4) est directement prise en compte dans la matrice $\mathbf{K}$, en remplaçant k_x (respectivement k_y) par $\alpha_x = fk_x$ (resp. $\alpha_y = fk_y$).

La transformation homogène ${}^{\mathcal{C}}\mathbf{T}_{\mathcal{W}}$ constitue les *paramètres extrinsèques* de la caméra. Ils permettent de positionner une caméra dans un environnement donné. Nous verrons par la suite comment cette grandeur peut être déterminée.

Les grandeurs α_x, α_y, θ, $x_{p_\mathcal{C}}$ et $y_{p_\mathcal{C}}$, contenues dans la matrice $\mathbf{K}$, correspondent aux *paramètres intrinsèques* d'une caméra. Ils représentent les caractéristiques internes d'une caméra, et ne dépendent pas de sa position dans l'espace. Ces paramètres sont généralement fournis par le constructeur, mais peuvent aussi être déterminés par des méthodes de calibration [Tsai 86, Heikkilä 00].

Dans certaines applications, il est nécessaire de prendre en compte le fait que la projection physique n'est pas réellement linéaire comme présenté en (1.3), où la caméra est supposée parfaite. Des paramètres de distorsion doivent alors être rajoutés aux paramètres intrinsèques [Devernay 95, Zhang 95a, Rémy 98]. Dans notre cas, nous supposerons que ces distorsions sont négligeables.

De nombreux autres modèles de caméra ont été proposés dans la littérature. Citons par exemple les modèles para-perspectif, ortho-perspectif ou encore parallèle. Le lecteur intéressé trouvera dans [Lingrand 99, Hartley 00] des illustrations de ces différentes modélisations qui sont des simplifications du modèle perspectif. Dans le reste de ce manuscrit, nous considérons le modèle général que nous avons présenté précédemment.

D'autres types de capteurs vision sont aujourd'hui également utilisés. Ainsi, certaines caméras possèdent une distribution fovéale en log-polaire des pixels afin de disposer d'une meilleure définition sur les zones d'attention [Bernardino 99]. Les caméras omni-directionnelles permettent d'obtenir des vues de 360 degrés, grâce à un miroir placé au-dessus de la caméra [Baker 99]. Du type du miroir utilisé (paraboloïque, hyperboloïque, elliptique) dépendent les relations de projection. Dans les travaux présentés ici, nous utilisons les caméras perspectives classiques.

1.1.3 Géométrie de deux images

Si l'on ne possède pas de connaissance *a priori* sur l'environnement observé, une image prise par une caméra fournit peu d'information sur la géométrie de la scène. À partir de deux images, il est possible de déduire non seulement la structure $3D$ de la scène, mais aussi le mouvement entre les deux caméras qui ont acquis ces prises de vue. Comme nous allons le voir dans cette partie, ces informations sont déduites des primitives mises en correspondance dans les deux images. La section 1.2 rappellera comment peut s'effectuer cette mise en correspondance.

1.1.3.1 La géométrie épipolaire

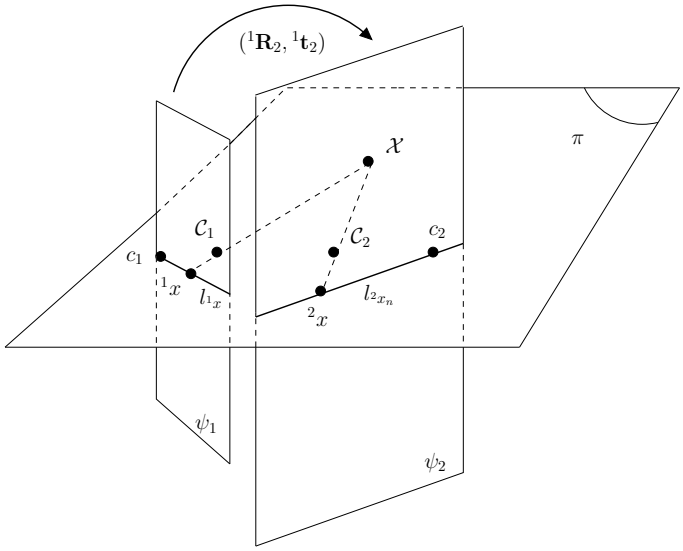


FIG. 1.3: Illustration de la géométrie épipolaire

Considérons deux caméras observant une même scène $3D$, comme sur la figure 1.3. Conformément à la définition d'une caméra que nous avons donnée, un centre de projection $\mathcal{C}_i$, un repère orthonormé $\mathcal{F}_i$ et des paramètres intrinsèques $\mathbf{K}_i$ sont associés à chacune des deux images ψ_i.

Soit $\left({}^1\mathbf{R}_2 \;\; {}^1\mathbf{t}_2\right)$ la transformation rigide permettant de passer de la pose de la première caméra $\mathcal{F}_1$ vers celle de la deuxième caméra $\mathcal{F}_2$. Au point $\mathcal{X}$ de la scène sont attribuées les coordonnées homogènes ${}^1\mathbf{X}$ dans le premier repère et ${}^2\mathbf{X}$ dans le deuxième. Ces deux vecteurs vérifient :

$$ {}^2\mathbf{X} = {}^2\mathbf{T}_1 {}^1\mathbf{X} $$

Le principe de la géométrie épipolaire est facilement déduit de la figure 1.3. Sur ce schéma, le point $\mathcal{X}$ est projeté dans la vue ψ_1 (respectivement ψ_2) en ${}^1x$ (resp. ${}^2x$). π est le plan généré par les points $\mathcal{C}_1$, $\mathcal{C}_2$ et le point de la scène $\mathcal{X}$. Il est clair que ${}^1x$ et ${}^2x$, intersections des lignes $(\mathcal{C}_1\ \mathcal{X})$ et $(\mathcal{C}_2\ \mathcal{X})$ avec les deux plans images, appartiennent à ce même plan. Les points c_1 et c_2, appelés *épipoles*, représentent l'intersection de la droite $(\mathcal{C}_1\ \mathcal{C}_2)$ avec les deux plans images. c_1 (respectivement c_2) correspond à la projection sur le premier (resp. deuxième) plan image du centre de projection de la deuxième (resp. première) caméra.

Si l'on considère l'image ψ_1, $\mathcal{X}$ appartient à la ligne de vue $\left({}^1x\ \mathcal{C}_1\right)$. De ce fait, le point correspondant dans la deuxième image est nécessairement le projeté d'un des points de cette droite. La droite l_{2x} dite *épipolaire*, intersection du plan π avec le plan image, représente l'ensemble des correspondants possibles de ${}^1x$ dans la deuxième vue. La géométrie épipolaire stipule ainsi que le correspondant d'un point d'une image appartient à la ligne épipolaire associée dans la deuxième image. Elle correspond à une relation point à ligne qui peut aussi être déduite directement de manière mathématique, comme présenté dans la section suivante.

1.1.3.2 Les matrices fondamentales et essentielles

La contrainte épipolaire s'appuie sur la coplanarité des points $\mathcal{C}_1$, $\mathcal{C}_2$, ${}^1x$, et ${}^2x$. En géométrie euclidienne, la coplanarité de 3 vecteurs $\overrightarrow{v_1}$, $\overrightarrow{v_2}$ et $\overrightarrow{v_3}$ est vérifiée si leur produit mixte est nul, soit $[v_1, v_2, v_3] = v_1(v_2 \wedge v_3) = 0$. Nous en déduisons :

$$ \overrightarrow{\mathcal{C}_1{}^1x} \left(\overrightarrow{\mathcal{C}_1\mathcal{C}_2} \wedge \overrightarrow{\mathcal{C}_2{}^2x} \right) = 0 $$

En choisissant le repère rattaché à la première caméra comme repère de référence, il vient :

$$
\begin{aligned}
{}^1\mathbf{x}_n^\top . \left(\overrightarrow{\mathcal{C}_1\mathcal{C}_2} \wedge \overrightarrow{\mathcal{C}_2{}^2x} \right) &= 0 \\
{}^1\mathbf{x}_n^\top . \left({}^1\mathbf{t}_2 \wedge \overrightarrow{\mathcal{C}_2{}^2x} \right) &= 0 \\
{}^1\mathbf{x}_n^\top . \left({}^1\mathbf{t}_2 \wedge {}^1\mathbf{R}_2 {}^2\mathbf{x}_n \right) &= 0 \\
{}^1\mathbf{x}_n^\top . \left(\left[{}^1\mathbf{t}_2 \right]_\times {}^1\mathbf{R}_2 {}^2\mathbf{x}_n \right) &= 0 \\
{}^1\mathbf{x}_n^\top \ \mathbf{E}^\top \ {}^2\mathbf{x}_n &= 0 \\
{}^2\mathbf{x}_n^\top \ \mathbf{E} \ {}^1\mathbf{x}_n &= 0
\end{aligned}
$$

où $[\mathbf{v}]_\times$ est la matrice antisymétrique associée au vecteur $\mathbf{v}$. La matrice $\mathbf{E}_{3\times3}$ est appelée *matrice essentielle*. Elle exprime la géométrie épipolaire dans le cas de projetés exprimés en mètre. Si les

paramètres intrinsèques ne sont pas connus, on travaille alors avec la *matrice fondamentale* $\mathbf{F}_{3\times3}$, qui est telle que :

$$\mathbf{F} = \mathbf{K}_2^{-\top}\mathbf{E}\mathbf{K}_1^{-1} \quad \text{et} \quad {}^2\mathbf{x}_p^\top \mathbf{F}\, {}^1\mathbf{x}_p = 0$$

Géométriquement, la matrice $\mathbf{F}$ définit l'ensemble des droites de ψ_2 passant par c_2. En particulier, ${}^1\mathbf{x}_p^\top \mathbf{F}^\top$ représente la ligne épipolaire à laquelle doit appartenir le projeté correspondant dans la deuxième image. Cette forte contrainte peut être utilisée par exemple pour faciliter une opération de mise en correspondance entre deux images [Zhang 94].

La matrice fondamentale est de rang 2, car elle correspond à une transformation projective de $\mathbb{P}^2$ dans $\mathbb{P}^1$. Elle possède 7 degrés de liberté, car elle est connue à un facteur d'échelle près, et la contrainte non linéaire ($\det(\mathbf{F}) = 0$) lui retire un autre degré de liberté. Chaque correspondance de points apportant une équation linéaire, 7 correspondances de points suffisent dans le cas général pour la déterminer [Hartley 97a].

L'épipole c_2 (respectivement c_1) appartient à toutes les lignes épipolaires de la deuxième (resp. première) vue. De ce fait, $\mathbf{c}_2$ (resp. $\mathbf{c}_1$) appartient au noyau gauche (resp. droit) de $\mathbf{F}$:

$$\mathbf{c}_2^\top \mathbf{F} = 0 \text{ et } \mathbf{F}\mathbf{c}_1 = 0 \tag{1.7}$$

1.1.3.3 Homographie entre deux prises de vue

La définition de la matrice d'homographie telle qu'elle est présentée ici provient de l'article de référence de Faugeras [Faugeras 88].

Considérons un plan de la scène, noté π, défini dans le repère de la caméra $\mathcal{F}_1$ par sa normale ${}^1\vec{\mathbf{n}} = (a\ b\ c)$, et la distance d_π au centre de projection de cette caméra. Un point de la scène ${}^1\mathbf{X} = (X_1\ Y_1\ Z_1)$ appartenant à π vérifie ainsi :

$${}^1\mathbf{n}^\top {}^1\mathbf{X} = d_\pi \tag{1.8}$$

En combinant (1.1) et (1.8), on obtient :

$${}^2\mathbf{X} = \left({}^2\mathbf{R}_1 + \frac{{}^2\mathbf{t}_1 {}^1\mathbf{n}^\top}{d_\pi} \right) {}^1\mathbf{X}$$

Or ${}^1\mathbf{X} = Z_1 {}^1\mathbf{x}_n$, d'où :

$$\begin{aligned} Z_2 {}^2\mathbf{x}_n &= \left({}^2\mathbf{R}_1 + \tfrac{{}^2\mathbf{t}_1 {}^1\mathbf{n}^\top}{d_\pi} \right) Z_1 {}^1\mathbf{x}_n \\ {}^2\mathbf{x}_n &= \tfrac{Z_1}{Z_2} \left({}^2\mathbf{R}_1 + \tfrac{{}^2\mathbf{t}_1 {}^1\mathbf{n}^\top}{d_\pi} \right) {}^1\mathbf{x}_n \\ {}^2\mathbf{x}_n &\propto {}^2\mathbf{H}_{n_1} {}^1\mathbf{x}_n \end{aligned} \tag{1.9}$$

La matrice ${}^2\mathbf{H}_{n_1}$, appelée *matrice d'homographie*, correspond à un changement de base dans l'espace projectif $\mathbb{P}^2$. Contrairement à la matrice essentielle, c'est une relation point à point entre les deux vues. Elle correspond à la composition de deux changements de base : l'un du plan image de la première caméra vers le plan de référence π, et un autre de ce plan vers le plan image de la deuxième caméra. ${}^2\mathbf{H}_{n_1}$ est une matrice 3×3. Elle ne peut être déterminée qu'à un facteur d'échelle près ; elle a donc huit degrés de liberté. Chaque couple de points donnant deux équations, quatre correspondances sont nécessaires pour estimer une telle matrice.

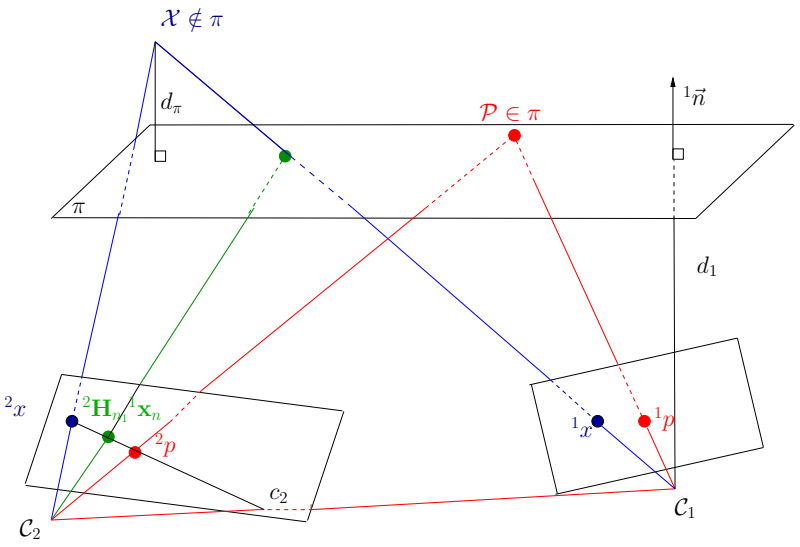


FIG. 1.4: Illustration de l'homographie entre deux vues

Si les points sont exprimés en coordonnées pixelliques, une relation analogue met en jeu la matrice ${}^2\mathbf{H}_{p_1}$ qui peut être déduite de la matrice ${}^2\mathbf{H}_{n_1}$, avec la relation :

$$ {}^2\mathbf{H}_{p_1} = \mathbf{K}_2 {}^2\mathbf{H}_{n_1} \mathbf{K}_1^{-1} \quad \text{et} \quad {}^2\mathbf{x}_p \propto {}^2\mathbf{H}_{p_1} {}^1\mathbf{x}_p \tag{1.10} $$

La relation (1.9) n'est valide que pour les points $\mathcal{X} \in \pi$. Il est cependant possible de définir une relation entre les deux projetés d'un point $3D$ n'appartenant pas à ce plan de référence. Reprenons pour cela l'équation générale :

$$ Z_2 {}^2\mathbf{x}_n = Z_1 {}^2\mathbf{R}_1 {}^1\mathbf{x}_n + {}^2\mathbf{t}_1 $$

D'après la formulation de l'homographie (1.9), et comme le point considéré n'appartient pas à π, on a :

$$ \begin{aligned} Z_2 {}^2\mathbf{x}_n &= Z_1 {}^2\mathbf{H}_{n_1} {}^1\mathbf{x}_n - \tfrac{Z_1}{d_\pi} {}^2\mathbf{t}_1 {}^1\mathbf{n}^\top {}^1\mathbf{x}_n + {}^2\mathbf{t}_1 \\ \tfrac{Z_2}{Z_1} {}^2\mathbf{x}_n &= {}^2\mathbf{H}_{n_1} {}^1\mathbf{x}_n + \left(\tfrac{1}{Z_1} - \tfrac{1}{d_\pi} {}^1\mathbf{n}^T {}^1\mathbf{x}_n \right) {}^2\mathbf{t}_1 \end{aligned} $$

En notant $d = d_\pi - {}^1\mathbf{n}^T (Z_1 {}^1\mathbf{x}_n)$ la distance signée du point $\mathcal{X}$ à π, on obtient :

$$ \frac{Z_2}{Z_1} {}^2\mathbf{x}_n = {}^2\mathbf{H}_{n_1} {}^1\mathbf{x}_n + \frac{d}{Z_1 d_\pi} {}^2\mathbf{t}_1 \tag{1.11} $$

Là encore, cette relation peut être établie avec des points exprimés en pixel :

$$ \frac{Z_2}{Z_1} {}^2\mathbf{x}_p = {}^2\mathbf{H}_{p_1} {}^1\mathbf{x}_p + \frac{d}{Z_1 d_\pi} \mathbf{c}_2, $$

où $\mathbf{c}_2 = \mathbf{K}_2 {}^2\mathbf{t}_1$. En notant $\beta = \frac{d}{Z_1 d_\pi}$, nous obtenons la relation :

$$ {}^2\mathbf{x}_p \propto {}^2\mathbf{H}_{p_1} {}^1\mathbf{x}_p + \beta \mathbf{c}_2 \tag{1.12} $$

Comme on le voit sur le schéma 1.4, la projection ${}^2\mathbf{H}_{n_1}\,{}^1\mathbf{x}_n$ revient à considérer que le point $\mathcal{X}$ appartient au plan. Un déplacement le long de la ligne épipolaire permet de trouver le bon correspondant. Le terme β est souvent appelé *parallaxe* dans la littérature [Boufama 95, Triggs 00a]. En géométrie projective orientée, le signe de β permet de déduire la position du point par rapport au plan de référence [Stolfi 91]. Il est important de noter que le terme β ne dépend que de paramètres exprimés dans le repère de la première caméra.

Si l'on connaît l'homographie entre les deux images ainsi que l'épipole, il est possible de déterminer ce *ratio* de profondeur (ici dans le cas de points exprimés en coordonnées pixelliques), en utilisant (1.12) :

$$\beta = -\frac{\left({}^2\mathbf{H}_{p_1}\,{}^1\mathbf{x}_p \wedge {}^2\mathbf{x}_p\right)^\top \left(\mathbf{c}_2 \wedge {}^2\mathbf{x}_p\right)}{\|\mathbf{c}_2 \wedge {}^2\mathbf{x}_p\|^2} \tag{1.13}$$

Notons enfin que chacune des deux formulations matricielles de la géométrie épipolaire (matrice fondamentale ou matrice d'homographie) peut être exprimée en fonction de l'autre. Ainsi, dans la seconde vue, le point correspondant à ${}^1x$ appartient à la ligne épipolaire l_2, ce qui peut s'écrire :

$${}^2\mathbf{x}_n^\top \mathbf{l}_2 = 0 \tag{1.14}$$

L'épipole c_2 et le point de coordonnées pixelliques ${}^2\mathbf{H}_{p_1}\,{}^1\mathbf{x}_p$ appartiennent aussi à cette ligne épipolaire, ce qui peut s'exprimer sous la forme :

$$\begin{aligned} {}^2\mathbf{x}_n^\top \left(\mathbf{c}_2 \wedge {}^2\mathbf{H}_{p_1}\,{}^1\mathbf{x}_p\right) &= 0 \\ {}^2\mathbf{x}_n^\top \left([\mathbf{c}_2]_\times\,{}^2\mathbf{H}_{p_1}\right)\,{}^1\mathbf{x}_p &= 0, \end{aligned}$$

ce qui nous permet de définir la transformation dont on déduit la matrice fondamentale à partir de la matrice d'homographie :

$$\mathbf{F} = [\mathbf{c}_2]_\times\,{}^2\mathbf{H}_{p_1} \tag{1.15}$$

Nous allons maintenant nous intéresser aux différentes méthodes permettant à partir des correspondances de points de déduire la nature du mouvement rigide entre ces deux vues de la scène.

1.1.3.4 Estimation du mouvement rigide

À partir de primitives en correspondance, l'estimation du mouvement rigide reliant les deux positions de la caméra passe par l'estimation de la géométrie épipolaire. Nombreuses sont les méthodes permettant de déterminer cette géométrie. Nous n'effectuerons ici qu'une présentation succincte des méthodologies employées. Nous renvoyons aux études comparatives de [Luong 96, Zhang 98] (pour la détermination des matrices fondamentales et essentielles) et de [Malis 00] (pour la matrice d'homographie) pour de plus amples détails. De plus nous nous limiterons au cas où les primitives considérées sont des points.

Comme il a déjà été mentionné, sept correspondances sont suffisantes pour calculer la matrice fondamentale de manière linéaire [LonguetHiggins 81]. Une normalisation des données permet d'obtenir une meilleure estimation de cette matrice [Hartley 97a]. Dans [Luong 96], Luong montre que la solution obtenue peut cependant s'avérer instable, et qu'elle tend à ramener les épipoles vers le centre de l'image. De plus, la contrainte de rang n'est généralement pas satisfaite.

Des méthodes de résolution non linéaires ont ensuite été proposées afin de prendre en compte les contraintes portant sur cette matrice au moyen de représentations adaptées. Le critère de

minimisation généralement utilisé correspond à la distance entre un projeté et sa ligne épipolaire associée [Deriche 94, Luong 96]. Ces méthodes nécessitent une bonne initialisation de la matrice fondamentale, initialisation qui est généralement obtenue par une approche linéaire. Les méthodes non-linéaires permettent ensuite de faire converger cette mesure initiale vers la solution optimale au sens du critère de minimisation.

Certaines méthodes déterminent la matrice fondamentale à partir d'une homographie et de l'épipole. L'intérêt de cette méthodologie est qu'indirectement, elle permet de contrôler le rang de la matrice fondamentale. Ainsi, dans [Boufama 95], un changement de repère dans l'image, associé à la contrainte de colinéarité entre c_2, ${}^2x$ et le point de coordonnées pixelliques ${}^2\mathbf{H}_{p_1}\,{}^1\mathbf{x}_p$ permet de déterminer simultanément l'homographie et l'épipole. La matrice fondamentale est alors obtenue en appliquant l'équation (1.15). Mais cette méthodologie possède des cas singuliers où la matrice d'homographie ne peut être estimée.

Dans [Shashua 94b], la matrice d'homographie est calculée de manière linéaire à partir de trois correspondances et des épipoles. Ces dernières sont déduites de l'équation (1.7), ce qui suppose donc que la matrice fondamentale est estimée préalablement. Dans [Couapel 95, Malis 00], la matrice d'homographie est déduite directement, sans nécessiter la connaissance de l'épipole, là aussi au moyen d'un changement de repère judicieux. Dans [Malis 00], il est montré que la deuxième méthode permet d'obtenir une matrice moins sensible aux mesures bruitées.

Il existe des configurations où la matrice fondamentale subit une réduction de rang. Entre autres, cette dégénérescence est observée si :
- le mouvement entre les deux positions de la caméra est une rotation pure ;
- toutes les primitives de la scène appartiennent à un même plan ;
- tous les points $3D$ de la scène se projettent sur une droite unique.

La matrice d'homographie est au contraire toujours définie. Si le mouvement entre les deux images est une rotation pure, la matrice d'homographie s'écrit alors ${}^2\mathbf{H}_{n_1} = {}^2\mathbf{R}_1$, ce qui revient à considérer que le plan de référence est le plan à l'infini. Si la scène observée est plane, la relation (1.9) est exacte. Si l'on a n correspondances, l'estimation de l'homographie revient à déterminer $n + 8$ inconnues avec un système de $3n$ équations (quatre correspondances entre les deux prises de vue sont alors suffisantes).

Une fois la géométrie épipolaire estimée, il est possible de déterminer le mouvement rigide reliant les positions des deux caméras. Ce mouvement, ainsi que d'autres informations sur la structure de la scène, peuvent être déduits de la matrice essentielle ou bien de la matrice d'homographie associée à des points exprimés en mètres (nous ne considérons pas ici connaître le modèle de la scène, ni aucun *a priori* sur sa structure).

Grâce à une décomposition SVD (décomposition en valeurs singulières) de la matrice essentielle, [Hartley 92], il est possible d'extraire la rotation ${}^2\mathbf{R}_1$, et la translation à un facteur d'échelle près $\frac{{}^2\mathbf{t}_1}{d_1}$. Ces mêmes paramètres peuvent être extraits de la matrice d'homographie (tout comme la normale ${}^1\vec{n}$ au plan π) [Faugeras 88, Zhang 95b]. Dans le cas général, deux solutions sont possibles. Cette ambiguïté peut être levée si [Faugeras 88] :
- une information *a priori* sur la scène est utilisée, comme une connaissance approximative de la normale au plan de référence, ou encore la connaissance de deux droites parallèles ;
- deux homographies relatives à deux plans différents sont estimées, et la solution commune

est retenue ;
- une troisième vue ψ_3 est utilisée, l'homographie ${}^3\mathbf{H}_{n_1}$ est estimée avec le même plan de référence. L'estimation de l'équation du plan avec ces deux homographies permet de lever l'ambiguïté.

En plus du mouvement partiel entre les deux images, l'homographie peut nous informer sur la structure de la scène, notamment sur [Malis 98] :
- le rapport r entre les distances d_1 et d_2 des centres de projection $\mathcal{C}_1$ et $\mathcal{C}_2$ au plan de référence π :

$$r = \frac{d_2}{d_1} = \det({}^2\mathbf{H}_{n_1}) = 1 + {}^1\mathbf{n}^\top {}^2\mathbf{R}_1 \frac{{}^2\mathbf{t}_1}{d_1} \tag{1.16}$$

- le rapport τ entre les profondeurs Z_2 et Z_1 d'un point $3D$:

$$\begin{cases} \tau = \frac{Z_2}{Z_1} = \frac{{}^1\mathbf{n}_1^\top {}^1\mathbf{x}_n}{{}^2\mathbf{n}^\top {}^2\mathbf{x}_n} r & \text{si } \mathcal{X} \in \pi \\ \\ \tau = \frac{Z_2}{Z_1} = \frac{\|[{}^2\mathbf{t}_1]_\times {}^2\mathbf{R}_1 {}^1\mathbf{x}_n\|}{\|[{}^2\mathbf{t}_1]_\times {}^2\mathbf{x}_n\|} & \text{si } \mathcal{X} \notin \pi \end{cases} \tag{1.17}$$

- le rapport ρ entre la profondeur Z_2 et la distance d_1 :

$$\begin{cases} \rho = \frac{Z_2}{d_1} = \frac{r}{{}^2\mathbf{n}^\top {}^2\mathbf{x}_n} & \text{si } \mathcal{X} \in \pi \\ \\ \rho = \frac{Z_2}{d_1} = \tau \frac{\|{}^2\mathbf{t}_1/d_1\|}{\|{}^2\mathbf{t}_1/Z_1\|} & \text{si } \mathcal{X} \notin \pi \end{cases} \tag{1.18}$$

avec :

$$\frac{{}^2\mathbf{t}_1}{Z_1} = \tau {}^2\mathbf{x}_n - {}^2\mathbf{R}_1 {}^1\mathbf{x}_n \tag{1.19}$$

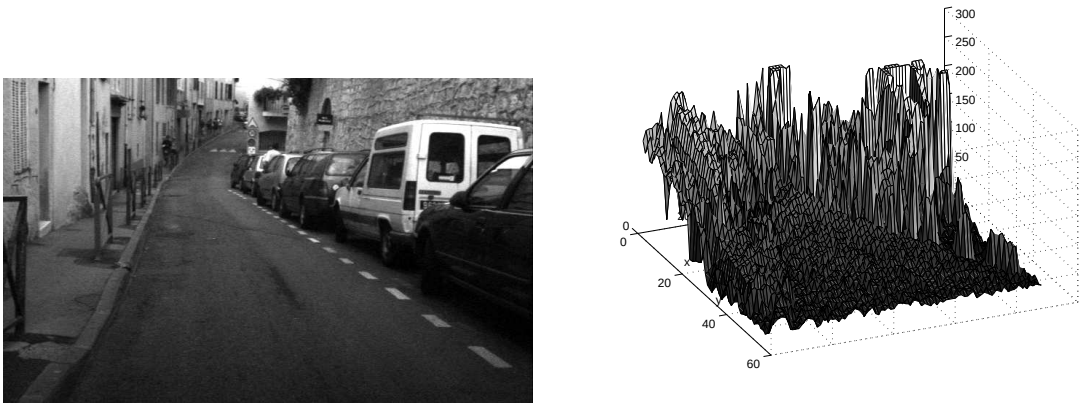
Ces relations nous seront utiles durant la phase de navigation comme nous le verrons dans le chapitre 4.

1.2 Description du contenu d'une prise de vue

Cette partie rappelle les traitements bas-niveaux permettant d'extraire des primitives d'une image acquise par une caméra. Nous verrons tout d'abord comment se présente une image d'un point de vue informatique, ce qui mettra en évidence la grande complexité de l'analyse d'images. Puis nous aborderons les méthodes permettant d'extraire d'une image des primitives dites *caractéristiques*. L'un des enjeux de cette extraction est de détecter des primitives pouvant être relocalisées dans d'autres vues de la même scène. Nous verrons donc ensuite comment ces primitives peuvent être suivies dans une séquence d'images, ou encore mises en correspondance entre deux vues.

1.2.1 Représentation des images

Une image acquise par une caméra en niveaux de gris correspond à une matrice de pixels. La valeur des pixels, comprise généralement entre 0 et 255, est une discrétisation de l'intensité lumineuse transmise par un point de la scène. Certains capteurs fournissent des images en couleurs RVB (rouge, vert et bleu) soit trois valeurs par pixels pour les trois couleurs rouge, vert et bleu. Nous nous focaliserons dans cette partie sur les images en niveaux de gris, et renvoyons vers l'ouvrage [Trémeau 04] pour plus d'informations sur les capteurs couleurs.



(a) (b)

FIG. 1.5: Comparaison entre notre vision d'une scène (a), et sa représentation numérique (b)

La figure 1.5 permet d'appréhender la difficulté du problème abordé en traitement d'images. La vue de gauche correspond à la vision que nous avons d'une scène. Les connaissances acquises par un long et complexe apprentissage nous permettent même à partir d'une seule vue d'imaginer la structure interne de la scène observée. L'image de droite présente quant à elle la matrice (sous-échantillonnée) de pixels associée à cette image, où le signal en niveau de gris est considéré comme une fonction. Il est clairement impossible même pour un humain d'en extraire rapidement une quelconque signification. C'est pourtant ce type d'information qui doit être traité ici.

On note dans la figure 1.5(b) de nombreux « pics », qui correspondent à de fortes variations locales du signal en niveaux de gris. Si l'on considère que la seule source de variation de ce signal est le mouvement de la caméra, on peut supposer que ces zones particulières dites d'intérêt restent localement identiques lors d'un déplacement du capteur vision. On peut donc espérer retrouver dans une nouvelle image une même distribution locale de la fonction d'intensité.

Se focaliser sur des zones d'intérêt dans l'image permet de diminuer très fortement la quantité de données à traiter. La partie suivante présente les méthodes de détection de ces zones d'intérêt.

1.2.2 Caractérisation des primitives visuelles

La nature de la scène observée permet de sélectionner les primitives les plus adéquates. Les environnements d'intérieur sont par définition très structurés. Visuellement, ils sont caractérisés par la présence de nombreux contours ou de lignes contrastées. D'un point de vue du signal, on peut caractériser ces primitives par une forte variation mono-dimensionnelle de l'intensité lumineuse. Les scènes d'extérieur sont quant à elles généralement assez texturées ; les primitives susceptibles d'être observées sont souvent assimilées visuellement à des coins, et correspondent à une forte variation bi-dimensionnelle du signal.

Comme nous désirons traiter des environnements extérieurs, nous utiliserons des points d'intérêts tant pour le traitement d'images que pour la navigation. Nous nous focaliserons donc sur ce type de primitives dans cet ouvrage.

Les méthodes de détection de points peuvent être regroupées en trois classes [Schmid 96]. Certaines s'appuient sur une extraction préalable des contours ; soit on recherche les points de courbures maximales le long des chaînes de contour, soit on effectue une approximation polygonale de ces contours pour localiser des points particuliers (comme des intersections ou des inflexions). D'autres méthodes travaillent directement sur le signal en niveau de gris. Un point de l'image est caractérisé par les valeurs de niveaux de gris sur un voisinage (ou support). Un point dit d'intérêt peut se caractériser comme une forte variation bi-directionnelle du signal dans ce voisinage. Cette forte variation se caractérise par deux fortes courbures du signal. Les différentes mesures proposées cherchent à localiser au mieux ces zones. Enfin, une autre stratégie consiste à utiliser un modèle théorique de la primitive que l'on cherche à détecter. Par exemple, un coin peut être caractérisé par son axe de symétrie, son ouverture, les niveaux de gris alentours, la position du coin et le flou.

Dans [Schmid 00], une étude comparative de six détecteurs basés sur l'étude du signal est effectuée en se basant sur une mesure de répétabilité. Ce critère vérifie si la détection est invariante aux changements de conditions de prise de vue : changement des paramètres intrinsèques et/ou extrinsèques de la caméra, et changements des conditions d'illumination. Les images considérées sont des plans texturés. Dans la majorité des cas, le détecteur de Harris [Harris 88] donne les meilleurs résultats. Dans [Tissainayagam 04], quatre détecteurs sont confrontés à des séquences vidéo sans mouvement. La variation du signal est alors la conséquence du bruit d'acquisition et des changements naturels des conditions d'illumination. Les séquences concernent des environnements intérieurs et extérieurs. Les critères considérés sont la stabilité du coin détecté, ainsi que la précision de sa localisation. Les meilleurs résultats sont obtenus avec le détecteur de Harris et celui de Kanade-Lucas-Tomasi, ou KLT [Shi 94] (notons que ce dernier n'a pas été testé dans [Schmid 00]).

Nous renvoyons le lecteur à ces papiers pour de plus ample détails sur les détecteurs de points. Nous nous concentrerons dans cette partie sur les deux détecteurs les plus utilisés, à savoir le détecteur de Harris, et le détecteur KLT, qui comme nous allons le voir sont très semblables.

Détecteur de Harris et KLT

Un point d'intérêt est caractérisé par une forte variation bi-directionnelle du signal. De ce fait le niveau de gris associé à un point d'intérêt diffère fortement des valeurs des autres pixels du voisinage. Par exemple, si $\mathbf{x}_p = (x, y)$ est un point d'intérêt, et $I(\mathbf{x}_p)$ le niveau de gris en ce point, une mesure d'autocorrélation telle que :

$$E(\mathbf{x}_p) = \sum_{u,v \in \mathcal{W}} w_{u,v} \left[I(x+u, y+v) - I(\mathbf{x}_p) \right]^2 \tag{1.20}$$

doit fournir une forte valeur. $\mathcal{W}$ désigne une fenêtre autour du point, et $w_{u,v}$ une pondération pour que la contribution de chaque pixel de la fenêtre soit dépendante de la distance au point d'intérêt. Un développement de Taylor de $I(x+u, y+v)$ donne :

$$I(x+u, y+v) = I(\mathbf{x}_p) + uI_x + vI_y + \mathcal{O}(u^2, v^2),$$

où I_x et I_y correspondent aux dérivées premières du signal. La mesure d'autocorrélation devient alors :

$$E(\mathbf{x}_p) = \sum_{u,v \in \mathcal{W}} w_{u,v} \left(u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2 \right)$$

Nous nous retrouvons avec la définition d'une quadratique, qui peut encore s'écrire sous la forme matricielle :

$$E(\mathbf{x}_p) = \sum_{u,v \in \mathcal{W}} w_{u,v} \mathbf{x}_{\mathcal{W}}^{\top} \mathbf{M} \mathbf{x}_{\mathcal{W}}, \text{ avec } \mathbf{M} = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \text{ et } \mathbf{x}_{\mathcal{W}}^{\top} = (u, v).$$

Les valeurs propres de $\mathbf{M}$ correspondent aux courbures principales de la fonction d'autocorrélation E. Le détecteur de Harris est basé sur l'hypothèse qu'un point d'intérêt se situe aux maxima locaux de la matrice d'autocorrélation $\mathbf{M}$. Cette matrice étant symétrique, elle peut être diagonalisée en effectuant une rotation des deux axes de coordonnées. Les valeurs propres obtenues (λ_1 et λ_2 telles que $\lambda_1 \geq \lambda_2$) permettent de caractériser la zone considérée :

- $\lambda_1 = \lambda_2 = 0$: la zone sélectionnée est complètement uniforme ;
- $\lambda_1 > \lambda_2 = 0$: la zone correspond à un contour. Le vecteur propre associé à λ_1 est perpendiculaire à ce contour ;
- $\lambda_1 > \lambda_2 > \lambda_t$ (λ_t étant un seuil) : la zone contient une variation bi-dimensionnelle du signal. Elle correspond donc à un point d'intérêt.

Plutôt que de calculer les valeurs propres de $\mathbf{M}$, Harris utilise les relations existant entre une matrice et ses valeurs propres :

$$\begin{array}{rcl} \det(\mathbf{M}) & = & \prod_i \lambda_i \\ \text{tr}(\mathbf{M}) & = & \sum_i \lambda_i, \end{array}$$

et propose la mesure de force H d'un point $\mathbf{x}_p$:

$$H(\mathbf{x}_p) = \det(\mathbf{M}) - \alpha \text{trace}^2(\mathbf{M}),$$

où α est un scalaire positif. Un point d'intérêt est caractérisé par une mesure de force supérieure à un seuil H_t fixé.

Le bruit de mesure entraîne cependant un mauvais conditionnement du calcul des dérivées premières de la matrice $\mathbf{M}$. Dans [Schmid 00], il est proposé de remplacer le masque utilisé pour calculer les dérivées par la dérivée d'une gaussienne.

Le détecteur KLT [Shi 94] utilise la même matrice d'autocorrélation $\mathbf{M}$. Comme nous le verrons, cette relation est déduite de la conservation de l'intensité lumineuse au cours du temps et d'un critère de similarité analogue à l'équation (1.20). L'enjeu du détecteur KLT est de sélectionner des primitives qui peuvent être facilement suivies lorsque la caméra se déplace. Afin d'opérer un suivi correct, les deux valeurs propres de cette matrice doivent être grandes. De plus, pour assurer un bon conditionnement de la matrice $\mathbf{M}$, les deux valeurs propres doivent être du même ordre de grandeur. La valeur de λ_1 étant limitée par les valeurs en niveaux de gris possibles, une condition suffisante est donc que λ_2 soit suffisamment grand. Contrairement au détecteur de Harris, les valeurs propres λ_1 et λ_2 sont ici calculées. Un point d'intérêt est alors localisé lorsque la plus petite valeur propre des deux est supérieure à un seuil donné λ_t.

La figure 1.6 illustre sur un exemple les points obtenus avec les deux détecteurs. Nous pouvons observer sur ces deux images que tous les points ne correspondent pas à des coins réels. Notons de plus sur l'image traitée avec Harris que certains points obtenus sont très proches les uns des autres. Les deux détecteurs filtrent les points détectés en ne conservant que celui de meilleur score dans un zone donnée. La taille de la fenêtre dans le cas du détecteur de Harris est plus petite que celle utilisée dans KLT. Le choix de la taille de cette fenêtre influence indirectement la répartition des points détectés.

(a) (b)

FIG. 1.6: Les 200 meilleurs points détectés par Harris (a) et KLT (b)

Il s'avère que le détecteur de Harris est largement utilisé pour les applications d'appariement et d'indexation d'images, alors que le détecteur KLT est très souvent intégré dans des applications de suivi de primitives. Une des contributions de cette thèse étant l'élaboration d'un formalisme de suivi de primitives adapté aux tâches de navigation, nous avons opté pour le détecteur de KLT, afin d'extraire les primitives les plus robustes pour le suivi.

1.2.3 La mise en correspondance

La mise en correspondance d'images, ou *l'appariement d'images*, a pour objectif de déterminer les correspondances entre deux images d'une même scène. Ainsi, si l'on considère un point ${}^1\mathbf{x}_p$ d'une image, projection d'un point $\mathcal{X}$ de la scène observée, il est alors recherché dans la deuxième vue la projection ${}^2\mathbf{x}_p$ correspondante.

Une des difficultés de cette mise en correspondance est que généralement, ${}^2\mathbf{x}_p$ est recherché sans avoir la connaissance de $\mathcal{X}$. De plus, les conditions d'acquisition peuvent fortement varier entre les deux prises de vue : les paramètres intrinsèques et/ou extrinsèques de la caméra peuvent être modifiés, les conditions d'illumination peuvent évoluer, certains objets apparaissent dans la deuxième image, et d'autres disparaissent.

Les méthodes d'appariement d'images peuvent être classées suivant le type d'images qu'elles traitent. Certains algorithmes sont particulièrement adaptés aux images dites structurées [Gros 98]. Pour de telles images, les lignes ou contours sont les primitives adaptées. Si les images sont texturées, il est plus judicieux d'utiliser des points d'intérêt.

Les mesures de corrélation ont été très largement utilisées pour réaliser la mise en correspondance [Faugeras 92, Zhang 94]. Les supports des points dans les deux images sont comparées par des mesures telles que la SSD (*somme des différences au carré*), la ZNCC (*corrélation normalisée centrée*) etc. Ce type de mesure est performant si les conditions d'acquisition des deux images sont proches. En effet, si l'on ne possède aucune connaissance *a priori* sur le mouvement entre les deux prises de vue, une simple rotation suffit à faire échouer la mise en correspondance. De plus, la mise en correspondance par corrélation est très bruitée, causant de nombreux mauvais

appariements. Deriche [Deriche 94] propose de caractériser un point d'intérêt par la direction du gradient, la courbure et la disparité, afin d'augmenter la discrimination. Zhang [Zhang 94] propose un appariement robuste en estimant la géométrie épipolaire entre les deux images.

Les valeurs en niveaux de gris autour du point d'intérêt n'étant pas assez discriminantes pour assurer un bon appariement, certains travaux se sont focalisés sur la caractérisation du signal dans le voisinage d'un point. Dans [Schmid 97], une description locale des points est réalisée au moyen d'invariants photométriques. Ces grandeurs sont invariantes aux différentes transformations image (rotations, translations, et partiellement aux changements d'illumination et facteur d'échelle). Un appariement croisé s'effectue ensuite en calculant une distance de Mahalanobis entre les vecteurs caractéristiques. Dans [Lowe 04], une autre mesure invariante aux rotations, au facteur d'échelle et aux translations est proposée. Chaque point est décrit par sa localisation, son échelle, et les orientations des gradients sur son voisinage (les points caractérisés étant recherchés sur plusieurs niveaux de résolution de l'image).

Ce type de caractérisation présente une forte discrimination des primitives. De ce fait, elles sont parfaitement adaptées à l'indexation d'images [Schmid 97] et à la reconnaissance d'objets [Lowe 01]. Cependant, dans un contexte d'appariement d'images, de mauvaises mises en correspondance peuvent tout de même être effectuées, surtout si les images traitées contiennent des motifs récurrents.

Dans le cadre de notre application, la mise en correspondance est un étape primordiale, puisqu'elle fournira les informations de base pour la structuration de la mémoire visuelle du robot (présentée dans le chapitre 2), pour le suivi (3) et pour la navigation (4). Nous avons donc opté pour la méthode d'appariement proposée par Zhang [2] qui a le mérite de contrôler les appariements effectués par une estimation robuste de la géométrie épipolaire [Zhang 94]. Les différentes étapes de cet algorithme sont les suivantes :

1. **Appariement par corrélation :** la corrélation utilisée est la ZNCC. Elle est appliquée sur des fenêtres autour des points d'intérêt. Au dessus d'un certain seuil de corrélation, des couples de points sont appariés. À ce stade de l'algorithme, un point d'une image peut être en correspondance avec plusieurs primitives dans l'autre image.
2. **Contrôle des appariements par relaxation :** cette phase permet de contrôler la cohérence spatiale des appariements précédents : si un point ${}^1\mathbf{x}_{pi}$ de la première vue est apparié avec ${}^2\mathbf{x}_{pj}$ dans la deuxième image, alors les points dans un voisinage proche de ${}^1\mathbf{x}_{pi}$ doivent être en relation avec les points dans le voisinage de ${}^2\mathbf{x}_{pj}$. Une mesure de support d'un appariement est proposée, et un processus de relaxation permet de minimiser la somme des supports de l'ensemble des mises en correspondance.
3. **Détermination de la géométrie épipolaire :** la matrice fondamentale est ici estimée à partir de huit points mis en correspondance. Les primitives utilisées sont choisies par des tirages aléatoires. La validité de la matrice fondamentale est mesurée par la somme des distances des points à leur droite épipolaire associée. Celle qui minimise l'erreur de reprojection est sélectionnée, et les points trop éloignés de leur droite épipolaire sont rejetés.

[2] le logiciel est téléchargeable sur http ://www-sop.inria.fr/robotvis/personnel/zzhang/softwares.html

1.2.4 Le suivi de primitives

Contrairement aux algorithmes d'appariement d'images, les méthodes de suivi de primitives traitent une séquence d'images acquise avec la même caméra, généralement à cadence vidéo. De plus, si la fréquence d'acquisition est élevée, le mouvement rigide entre deux prises de vue consécutives peut être considéré comme faible. La recherche de la position d'une primitive dans une nouvelle image peut donc se limiter à une recherche dans un voisinage proche. La plupart des techniques se basent sur cet *a priori*. Nous nous placerons dans le même contexte. Si de forts déplacements peuvent être obtenus entre deux vues successives, une approche pyramidale [Espiau 02] peut alors être utilisée. La fusion de plusieurs mesures (comme une mesure de similarité et le mouvement apparent dans l'image) permet aussi de gérer ces forts déplacements [Arnaud 04].

Là encore, de nombreux algorithmes de suivi utilisent des mesures de corrélation [Giachetti 00, Chambon 02]. Une fenêtre de recherche est définie dans la nouvelle image, centrée autour de la position du point dans la vue précédente. Le point minimisant la mesure de corrélation par rapport à une référence est alors sélectionné comme la nouvelle position du point d'intérêt. Le suivi par corrélation est une méthode de recherche exhaustive ; tous les points candidats de la fenêtre de recherche sont testés. La référence choisie n'est généralement pas la mesure dans l'image précédente. Le choix de cette référence est particulièrement délicat [Matthews 04]. Si le suivi doit s'effectuer sur une longue séquence, la référence peut par le mouvement de la caméra ou de l'objet devenir obsolète. Parallèlement, une mise à jour trop fréquente de la référence peut entraîner un phénomène de dérive qui fera suivre un point différent de l'originel.

Au début des années 80, Shi, Tomasi et Kanade [Lucas 81, Shi 94] ont proposé un algorithme de suivi basé sur une approche différentielle de la minimisation d'un critère de similarité. Cette méthode, toujours utilisée de nos jours, s'est vu proposée de nombreuses améliorations [Plakas 00, Jin 01, Baker 04]. Ces différents travaux qui s'appuient toujours sur les principes de base de la méthode KLT nous assurent du bien-fondé du choix de cet algorithme pour effectuer le suivi. Cette partie est donc consacrée aux principes du KLT.

Soit une séquence d'images $I(\mathbf{x}_p, t)$, où $I(\mathbf{x}_p, t)$ représente l'intensité lumineuse associée au point image de coordonnées pixelliques $\mathbf{x}_p = (x, y)$ dans l'image acquise à l'instant t. On suppose que la variation du niveau de gris est uniquement due au mouvement de la caméra ou de l'objet $3D$ considéré dans la scène. De plus, en considérant que la fréquence d'acquisition des images est suffisamment élevée, nous pouvons supposer qu'entre deux prises de vue consécutives le déplacement des primitives projetées est faible, et que leur intensité mesurée reste constante, soit :

$$I(\mathbf{x}_p, t) = I(\delta(\mathbf{x}_p), t + \tau),$$

où $\delta(.)$ est le champ de mouvement spécifiant le mouvement de chaque point de l'image, et τ un incrément temporel. Si la taille du support est petite, le champ de mouvement entre deux images consécutives peut être représenté par une translation $\mathbf{d}$ telle que :

$$\delta = \mathbf{d} = \begin{bmatrix} d_x \\ d_y \end{bmatrix}$$

Les coordonnées des pixels sont centrées en $\mathbf{x_p}$. Si $\mathcal{W}$ définit une fenêtre autour de $\mathbf{x_p}$, alors tout point $\mathbf{x}_{p_\mathcal{W}}$ de ce voisinage est tel que :

$$I(\mathbf{x}_{p_\mathcal{W}}, t) = I(\mathbf{x}_{p_\mathcal{W}} + \mathbf{d}, t + \tau)$$

Le modèle du mouvement n'étant pas exact, et l'image pouvant être bruitée, l'équation précédente n'est pas satisfaite exactement. On cherche alors le déplacement $\mathbf{d}$ qui va permettre de minimiser le résidu :

$$\varepsilon = \int\int_{\mathcal{W}} [I(\mathbf{x}_{p_\mathcal{W}} + \mathbf{d}, t+\tau) - I(\mathbf{x}_{p_\mathcal{W}}, t)]^2 w(\mathbf{x}_{p_\mathcal{W}}) d\mathbf{x}, \tag{1.21}$$

$w(\mathbf{x}_p)$ étant une gaussienne centrée en $\mathbf{x}_p$. En utilisant le développement de Taylor de 1^{er} ordre de $I(\delta(\mathbf{x}), t+\tau)$, nous obtenons le système linéaire suivant [Shi 94] :

$$\mathbf{Z}\mathbf{d} = \mathbf{a} \tag{1.22}$$

La matrice $\mathbf{Z}$ est de la forme :

$$\mathbf{Z} = \int\int_{\mathcal{W}} \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} w(\mathbf{x}_{p\mathcal{W}}) d\mathbf{x},$$

où g_x est la dérivée I_x convoluée avec une gaussienne. La matrice $\mathbf{Z}$ n'est rien d'autre que la matrice associée à la fonction d'autocorrélation du signal, sur laquelle se basent les deux détecteurs de points présentés ci-dessus. Le vecteur $\mathbf{a}$ dépend de la différence entre les deux images :

$$\mathbf{a} = \int\int_{\mathcal{W}} I_t \begin{bmatrix} g_x \\ g_y \end{bmatrix} w(\mathbf{x}_{p\mathcal{W}}) d\mathbf{x},$$

I_t désignant la dérivée temporelle du signal. Le système $\mathbf{d} = \mathbf{Z}^{-1}\mathbf{a}$ est généralement résolu par une minimisation itérative (comme la méthode Newton-Raphson par exemple).

La valeur du résidu ε permet de contrôler la qualité du suivi. Si la minimisation n'a pas convergée en un nombre fixé d'itérations, on peut considérer que le point est perdu, ou bien qu'on ne peut plus assurer son suivi de manière efficace.

Il est possible, lors de l'estimation de $\mathbf{d}$, d'atteindre un minimum local de la fonction de minimisation. La translation ne permet pas en effet de modéliser exactement le mouvement apparent. Plutôt que de se limiter à l'estimation de la translation par rapport à l'image précédente, il est possible de déterminer un modèle affine par rapport à une fenêtre de référence. Cette estimation peut se faire tout en gardant le même formalisme que celui présenté ci-dessus. De la même manière, la valeur de résidu obtenu permet de juger de la qualité du point suivi.

Dans [Plakas 00], une loi de rejet est utilisée pour détecter automatiquement les points qui ne peuvent être suivis correctement. Le résidu de l'équation (1.22) est utilisé dans un formalisme basé sur les $M-$estimateurs. Cette méthode permet de fixer un seuil ε_t au dessus duquel un point doit être rejeté.

Le modèle affine, en tant que modélisation d'une transformation géométrique, ne permet pas de prendre en compte les éventuels changements d'illumination, ce qui peut entraîner dans certaines configurations le rejet de points pourtant corrects. Jin et al. [Jin 01] proposent d'estimer dans le même formalisme deux paramètres représentant le changement de contraste et de luminance. Là aussi, une loi de rejet permet d'éliminer les mauvais points. Dans [Gouiffès 04], l'estimation des paramètres de réflexion de l'objet considéré permet de pouvoir utiliser ce type de suivi sur des objets non lambertiens.

Cependant, l'estimation du modèle affine, voire des paramètres de changement d'illumination, augmente considérablement les temps de traitement. Certes, ces temps restent convenables lorsqu'on effectue seulement une tâche de suivi ou quand le nombre de points suivis est limité. Dans les applications où la phase de suivi n'est qu'une tâche préliminaire avant un traitement de plus haut niveau, les temps séparant l'acquisition de deux images deviennent alors trop importants. Nous ne pouvons plus alors assurer qu'entre deux prises de vue le déplacement inter-image est faible, hypothèse qui est à la base de cette méthode de suivi. De ce fait, nous limiterons dans nos expériences le suivi à l'estimation du champ de mouvement entre deux vues successives. Pour éviter l'effet de dérive, nous estimerons le modèle affine périodiquement (plus de détails seront donnés dans le chapitre 3).

De plus, les articles traitant de lois de rejet se placent dans un cadre où le suivi est opéré sur une séquence enregistrée. Le rejet des mauvais points est réalisé à partir du résidu mesuré dans la dernière image. Cela signifie qu'un mauvais point peut être suivi et conservé jusqu'à la fin de la séquence. Dans notre application, les points suivis sont utilisés pour déterminer la loi de commande d'un robot. Nous ne pouvons donc pas nous permettre d'attendre la dernière image pour rejeter les mauvais points, puisque ceux-ci pourraient entraîner l'échec de la phase de navigation.

Il est à noter que d'autres modélisations du mouvement ont été proposées dans la littérature, afin de suivre des surfaces plus importantes qu'une simple fenêtre autour d'un point. La considération par exemple d'un plan de la scène permet de pouvoir utiliser l'hypothèse de conservation de l'information lumineuse. Dans ce contexte, une estimation d'une transformation affine voire homographique permet de mettre à jour la position des primitives suivies [Hager 98a, Jurie 02, Benhimane 04].

1.3 Conclusion

Ce chapitre a présenté la modélisation projective d'une caméra perspective. Les relations reliant les projetés de points dans deux prises de vue ont été décrites, et deux formulations matricielles de la géométrie épipolaire, à savoir la matrice fondamentale et la matrice d'homographie, ont été rappelées.

L'estimation de ces deux matrices peut s'effectuer à partir d'informations directement extraites des images. La matrice fondamentale présente cependant une dégénérescence dans de nombreuses configurations (scène plane, mouvement de rotation pur, etc). C'est pourquoi nous avons choisi d'utiliser la matrice d'homographie afin de représenter la géométrie épipolaire, puisque cette dernière n'est pas dégénérée dans ces cas de figure.

Des informations sur la structure de la scène peuvent être estimées à partir des matrices fondamentales et d'homographie. Nous avons vu aussi qu'une estimation du mouvement partiel peut être obtenue. Ces informations seront utiles pour la représentation et l'organisation de l'environnement de navigation (chapitre 2), pour la mise à jour des primitives visibles dans le module de suivi durant la navigation (chapitre 3), mais aussi pour définir les mouvements du système robotique (chapitre 4).

La deuxième partie de ce chapitre a abordé les techniques permettant d'analyser une image en niveau de gris. La localisation de points d'intérêt dans une image s'effectue très souvent par le

détecteur de Harris ou le détecteur KLT. Nous avons vu que ces deux techniques sont basés sur les mêmes principes. Les points obtenus avec le deuxième détecteur sont généralement mieux répartis dans les images. De plus, cet algorithme est très souvent employé par les applications effectuant un suivi de primitives. C'est pourquoi nous utiliserons cet algorithme pour détecter des points dans les images. L'appariement d'images a été abordé par de très nombreux travaux, pour différents types de scènes observées, ou différents systèmes de vision. Si les points automatiquement mis en correspondance doivent être utilisés pour définir les mouvements d'un système robotique, un grand soin doit être consacré à l'élimination d'éventuelles mauvaises mises en correspondance. C'est pourquoi nous avons choisi d'utiliser l'algorithme proposé et mis à disposition par Zhang. L'élimination des mauvais appariements est effectué en estimant la géométrie épipolaire entre les deux prises de vue.

Enfin, ce chapitre a présenté l'algorithme de suivi que nous proposons d'utiliser pour suivre des points d'intérêt au cours des mouvements du robot. L'algorithme de KLT est exploité dans de nombreuses applications robotiques basées vision. Cet algorithme est basé sur la conservation au cours du temps de l'information lumineuse. Cette hypothèse permet d'utiliser une approche différentielle pour suivre des points dans des séquences d'images.

Le contrôle de la validité des points suivis est un problème crucial du suivi des primitives. L'estimation de la translation inter-image n'est pas toujours suffisante, et il est alors nécessaire de comparer le signal au voisinage du point par rapport à une fenêtre de référence, via l'estimation d'un modèle affine. Nous verrons dans le chapitre 3 comment cette fenêtre de référence peut être choisie et mise à jour automatiquement.

Clairement, lors d'une tâche de navigation, de nombreuses primitives peuvent être amenées à sortir du champ de vision de la caméra, alors que d'autres peuvent rentrer dans celui-ci. Les techniques de suivi supposent que les primitives suivies restent dans la zone de vision de la caméra. Le chapitre 3 montrera comment l'utilisation des informations déduites de la géométrie épipolaire peut permettre de détecter l'apparition de nouvelles primitives d'intérêt, sans pour autant effectuer une recherche exhaustive de points sur toute l'image nouvellement acquise par la caméra, tout en s'affranchissant de la connaissance d'un modèle $3D$ de l'environnement de navigation.

Chapitre 2

Représentation d'un environnement de navigation

Ce chapitre est consacré à la représentation interne d'un espace de navigation, et aux méthodes de localisation associées. La première partie s'intéresse aux diverses représentations utilisées dans la littérature. De cette étude, nous justifierons les choix effectués, à savoir la mise en place d'une mémoire visuelle de l'environnement. Nous montrerons ensuite comment la mémoire visuelle peut être structurée, de façon à mettre aisément en relation la position initiale du robot avant son déplacement, et la position que celui-ci doit atteindre une fois la tâche de navigation effectuée. Nous finirons, enfin par une étude critique de notre formalisme, et par une étude des améliorations pouvant encore lui être apportées.

2.1 État de l'art

La problématique de la représentation d'un environnement de navigation a été abordée par de très nombreux travaux depuis le début des années 70. La synthèse que nous proposons ici ne traitera donc que des principaux formalismes qui ont été envisagés dans la littérature.

Au début des années 70, les avancées en vision par ordinateur, et les grands espoirs posés sur les puissances de calcul sans cesse grandissantes des machines ont amené les chercheurs à considérer que le robot peut « facilement » construire une carte de son environnement de navigation. Si l'on connaît le modèle de la scène, il est possible de projeter dans le référentiel associé au robot les obstacles qu'il doit éviter durant son déplacement (des murs, ou des objets statiques). Les recherches portaient alors principalement sur la planification de la trajectoire du robot dans cet espace particulier. Des capteurs tels que le sonar, le laser, ou un système stéréoscopique, peuvent ensuite être utilisés pour contrôler le bon suivi de la trajectoire précalculée.

Malheureusement, la reconstruction d'un environnement n'est pas une tâche aisée ; encore aujourd'hui, énormément de travaux abordent cette problématique [Laveau 96, Hartley 00, Triggs 00b]. En effet, des erreurs de reconstruction peuvent être engendrées par divers facteurs (les mesures des capteurs sont bruitées, la modélisation du robot est rarement parfaite, le robot peut glisser sur le sol, . . .).

C'est pourquoi, depuis le début des années 90, de nombreuses approches proposent de s'affranchir de la reconstruction $3D$ de la scène. Puisqu'il existe une relation directe entre la position du robot et les observations, ou signatures, de ses capteurs extéroceptifs, un environnement peut directement être défini dans l'espace des observations des capteurs. Le modèle de la scène n'est alors plus explicitement nécessaire.

Ainsi, deux grandes familles de représentation d'un espace de navigation se distinguent :
- *les représentations géométriques :* le modèle est supposé connu, ou construit par le robot. L'espace de navigation est alors exprimé dans un repère euclidien de référence ou bien projeté dans un référentiel propre au robot.
- *les représentations topologiques :* l'environnement est modélisé par les mesures fournies par les capteurs lors d'une phase d'apprentissage. Une discrétisation de ces informations est alors organisée dans un graphe topologique.

2.1.1 Représentation géométrique

Cette section débute par la présentation des méthodes réalisant une projection de l'environnement dans l'espace des configurations du système robotique. Des descriptions plus complètes sont proposées dans les ouvrages [Latombe 93, Siegwart 04]. Les méthodes représentant l'environnement comme un ensemble d'amers référencés dans un repère lié à la scène sont ensuite décrites.

2.1.1.1 Projection de l'environnement dans l'espace des configurations

Supposons, ici, que nous connaissons parfaitement le modèle de la scène. Supposons, aussi, que nous avons une confiance aveugle dans la modélisation du robot et de ses capteurs. La principale contrainte qu'il reste alors à satisfaire est l'évitement d'obstacles (un obstacle pouvant être un mur ou un objet de la scène).

L'espace de travail du robot peut être différent de son espace des configurations. L'espace de travail correspond à l'espace dans lequel se déplace le robot. Il peut être $3D$ si le robot peut se mouvoir dans l'espace tri-dimensionnel, ou $2D$ si le robot se déplace sur le sol. L'espace des configurations prend, quant à lui, en compte tous les degrés de liberté du robot. Par exemple, si un robot peut se translater suivant les trois axes du repère classique $3D$ tout en ayant la possibilité de pivoter sur chacun de ces axes, son espace de configuration est de dimension 6. L'espace des configurations est généralement de plus grande dimension que l'espace de travail.

La prise en compte des configurations interdites du robot durant la planification est facilitée si l'on projette les obstacles dans l'espace des configurations du robot. Un obstacle dans l'espace euclidien est appelé un $\mathcal{C}-$*obstacle* dans l'espace des configurations, et correspond à l'ensemble des configurations pour lesquelles l'obstacle et le robot sont en contact.

Dans un tel formalisme, un environnement de navigation correspond à l'ensemble des configurations libres du robot noté $\mathcal{C}_{libre}$. Dans la littérature, deux approches sont proposées pour représenter l'espace libre : la génération de cartes, et la décomposition en cellules libres.

La génération de cartes est souvent appelée *skeletonization method* ou *roadmap method* dans la littérature anglosaxonne. L'espace libre est représenté par une succession de courbes monodimensionnelles formant un squelette ou une carte de l'environnement noté $\mathcal{R}$ [Canny 88]. Les

éléments de $\mathcal{C}_{libre}$ n'appartenant pas à $\mathcal{R}$ doivent pouvoir être facilement connectés à la carte. De nombreuses stratégies ont été proposées pour réaliser ces cartes.

Les *graphes de visibilité* relient tous les sommets délimitant $\mathcal{C}_{libre}$ si le segment associé ne traverse pas un $\mathcal{C}$*-obstacle* [LozanoPerez 79, Ghosh 87, Laumond 87, Arikan 01]. C'est la technique la plus ancienne [Nilsson 69]. Ce type de carte est difficile à établir pour des espaces de configuration supérieurs à 2, ou si les $\mathcal{C}$-obstacles ne sont pas polygonaux.

Les méthodes exploitant les *diagrammes de Voronoï* cherchent au contraire à obtenir des courbes qui soient les plus distantes possibles des $\mathcal{C}$*-obstacles*. Ces diagrammes sont obtenus par des approches de rétroaction [Yap 85]. Dans [Yuen 02], le diagramme de Voronoï de l'environnement permet de définir les positions où le robot doit acquérir des images panoramiques afin de décrire visuellement la scène.

La méthode des *cônes généralisées* (ou *freeway net* [Brooks 83]) est dédiée aux robots se déplaçant à la fois en translation et en rotation dans un espace de travail $2D$. Un cône correspond à un cylindre linéaire généralisé [Binford 71] dont l'axe (*spin*) est annoté par une description des orientations libres que le système robotique peut prendre quand son point de référence se déplace le long de ce spin. Les différents cônes sont connectés entre eux dans un réseau. Deux cônes qui s'intersectent sont reliés par un arc si les deux ensembles d'orientations libres ont une intersection non nulle au point de croisement des deux axes.

Les *cartes probabilistes* [Kavraki 96, Laumond 00] sont souvent utilisées pour des systèmes robotiques possédant de nombreux degrés de liberté. Elles s'affranchissent du calcul explicite de la représentation géométrique de l'espace libre. Cette carte est réalisée durant une phase d'apprentissage, en tirant aléatoirement des configurations. Un test de collision permet de savoir si l'attitude considérée appartient à l'espace libre. Si cela est le cas, la configuration est rajoutée à la carte. Les configurations proches de la carte sont connectées avec ce nouveau sommet si le segment engendré est inclus dans $\mathcal{C}_{libre}$.

La décomposition en cellules consiste à décomposer l'espace des configurations en une collection de régions disjointes. Ces cellules peuvent être libres (pas de contact avec les $\mathcal{C}$-obstacles) ou interdites au robot [Avnaim 88, Zhu 91, Thrun 96]. La géométrie des cellules doit être suffisamment simple pour que les déplacements intra et inter cellules puissent être obtenus très facilement. La décomposition en cellules peut être exacte ou approchée [Chazelle 87]. La première décrit totalement l'espace libre, mais sa génération s'avère très complexe. La deuxième (souvent appelée méthode des *quadtree* ou des *octree* suivant la dimension de l'espace de configuration) est en revanche obtenu beaucoup plus facilement, par un processus récursif. Notons que cette méthodologie est très souvent utilisée en synthèse d'images non seulement pour définir une trajectoire [Lamarche 04], mais aussi pour déterminer les primitives qui sont dans le champ de vision de la caméra virtuelle (ce qui permet de n'effectuer des opérations de projection que pour les primitives susceptibles d'être visibles).

2.1.1.2 Modèle tridimensionnel de la scène

Les approches présentées ci-dessous représentent l'environnement dans l'espace de travail. Les objets de la scène ne sont plus considérés explicitement comme des obstacles, mais plutôt comme des amers. La trace de ces amers dans les mesures effectuées par les capteurs extéroceptifs permet de générer un modèle local qui doit être mis en relation avec le modèle global de la scène.

Si la structure $3D$ de la scène est considérée comme connue *a priori*, la localisation du robot consiste à recaler le modèle local avec le modèle global de l'environnement. Dans [Dao 03], une technique linéaire permet de calculer rapidement la position du robot par rapport à un modèle d'une scène d'intérieur. Les correspondances entre les lignes du modèle et celles extraites de l'image permettent de déterminer l'orientation du robot par rapport au référentiel. La position du robot est ensuite déduite des correspondances de points. Pour que ce processus de localisation soit efficace, le robot doit déjà avoir une bonne estimation de sa position, obtenue par exemple par odométrie.

Dans [Cobzas 01], la localisation est déduite des surfaces planes observées. La position relative du robot est déduite des contraintes portant sur les normales des plans extraits de l'image et celles des plans du modèle. L'extraction des plans de l'image est facilitée par le système trinoculaire employé. La sélection des plans de la base s'effectue en comparant leur intensité moyenne. Une hypothèse forte est posée, puisque les conditions d'illumination de la scène sont supposées ne varier que très faiblement.

De nombreux travaux proposent de coupler les caméras avec d'autres capteurs, ce qui permet par exemple de choisir lors de la localisation le capteur le plus adéquat. Dans [Hayet 02], une caméra et un laser sont associés sur un système robotique se déplaçant en environnement intérieur. Le capteur de vision est employé lorsque le laser ne permet pas de fournir une bonne localisation. Les amers visuels considérés sont dans ce cas des zones planes et texturées, comme des affiches sur les murs, ou des portes. Connaissant la position absolue des ces amers dans le modèle de la scène, la localisation revient à estimer la pose de la caméra par rapport aux amers visibles.

Le bruit présent dans les mesures des capteurs peut occasionner une mauvaise détection des amers, et donc un bruit sur la localisation du robot. Ce bruit se caractérise par un décalage entre la position mesurée des amers et la position estimée par projection de la scène $3D$ dans l'espace du capteur. Pour compenser ce buit, une solution consiste à réitérer le processus de localisation jusqu'à la minimisation de l'erreur de projection. Une telle localisation incrémentale est utilisée dans [Cobzas 03]. Dans ce papier, une scène d'intérieur est observée par une caméra et un laser ; les amers considérés sont des lignes verticales. Une contrainte de cette méthode est qu'elle nécessite tout de même une initialisation manuelle de la localisation du robot. Ce dernier est donc capable de se localiser durant sa navigation, mais incapable de détecter sa position initiale.

Une autre approche consiste à associer à chaque amer détecté par le capteur une incertitude quant à sa position réelle. Dans [Ohya 98], un filtre de Kalman étendu est utilisé pour estimer la position du robot à partir des correspondances entre les lignes extraites de l'image et leurs positions estimées d'après la position du robot et le modèle $3D$.

Si le modèle de la scène n'est pas connu, le robot a pour tâche d'en effectuer une reconstruction. Cette phase est très souvent considérée comme un processus *hors-ligne* : un opérateur humain déplace le robot tandis que celui-ci enregistre les mesures de ces capteurs. Le traitement de ces données permet alors de reconstruire l'environnement, sans (trop) se soucier des temps d'analyse. Comme nous l'avons vu dans le chapitre 1, deux images permettent de capturer la structure de la scène observée ainsi que le mouvement entre les deux positions associées. Ces reconstructions locales doivent ensuite être réexprimées dans un même repère de référence. La propagation des erreurs de reconstruction est généralement minimisée par des techniques d'ajustement de rayons (ou *bundle adjustment* [Hartley 93, Laveau 96, Hartley 00, Triggs 00b]) ; nous renvoyons le lecteur à ces références pour plus de détails sur ce type de traitement.

Actuellement, de nombreuses méthodes réalisent simultanément la localisation du robot et la modélisation de l'environnement. La reconstruction est alors réalisée en-ligne, alors que le robot se déplace de manière autonome. Dans ces méthodes dites de SLAM (*Simultaneous Localization And Mapping*), des capteurs permettant d'obtenir directement la profondeur des amers de la scène (comme le laser ou le sonar) sont souvent utilisés [Victorino 00, Thrun 00]. Les capteurs de vision ont aussi été largement utilisés [DeSouza 02], avec des amers naturels ou artificiels [Borenstein 96]. Si le système robotique ne possède qu'une seule caméra, le mouvement du robot permet d'appréhender la structure $3D$ de la scène. L'utilisation de plusieurs caméras [Davison 98, Se 02] ou le couplage d'une caméra avec un autre capteur permet d'associer une profondeur à chaque amer visuel [Ohya 98, Cobzas 03].

Dans [Se 02], un système robotique doté d'une vision trinoculaire réalise la localisation grâce à l'appariement des points connus avec leurs positions prédites dans les nouvelles vues. La mise en correspondance est facilitée par les contraintes géométriques du capteur, et par les informations d'échelle et d'orientation fournies par la description locale des points [Lowe 04]. De nouvelles primitives sont détectées par une recherche dans l'image, et rajoutées ensuite au modèle. Dans [Davison 98, Se 02], l'incertitude sur la localisation des primitives et sur la position du robot est propagée par un filtre de Kalman étendu. De nombreuses autres techniques probabilistes et statistiques ont été proposées pour modéliser le bruit des mesures, comme par exemple les approches markoviennes [Simmons 95, Burgard 98], ou encore les localisations par la méthode de Monte Carlo [Dellaert 99, Thrun 00].

2.1.2 Représentation basée apparence

Contrairement aux méthodes présentées précédemment, les approches représentant l'environnement par son apparence n'ont pas explicitement besoin du modèle de la scène. La localisation s'effectue en comparant directement les informations sensorielles avec celles mémorisées. Ces informations sont généralement organisées sous la forme d'un graphe topologique. Suivant les méthodes utilisées,

- le graphe stocke des séquences d'images ;
- chaque noeud du graphe correspond à une vue de la scène ;
- un noeud du graphe correspond à un lieu de la scène, et donc regroupe plusieurs images.

Dans ce type de représentation, il est important de voir que la localisation effectuée n'est pas nécessairement quantitative. Une localisation qualitative est souvent recherchée, c'est-à-dire que l'on recherche les vues sauvegardées qui sont les plus proches de ce qu'observe actuellement la caméra du robot.

L'information réellement stockée dans la base correspond aux descripteurs utilisés pour la localisation : les images en niveaux de gris, un ou plusieurs descripteurs par image, ou encore un descripteur pour un ensemble d'images.

2.1.2.1 Description par séquences d'images

La description d'un environnement de navigation par un ensemble de séquences d'images est clairement orientée vers la navigation autonome. Si le robot connaît sa position dans une des séquences, il peut alors « rejouer » le reste de la séquence pour se déplacer. Les scènes considérées

sont souvent d'intérieur. Néanmoins, Matsumoto et *al.* proposent de traiter des environnements extérieurs avec un système stéréoscopique [Matsumoto 00b]. Mais les déplacements du robot sont limités à des translations le long de l'axe optique.

Les séquences constituent dans le graphe d'adjacence les arcs qui permettent de relier les différents nœuds (chaque nœud étant associé à un emplacement). L'acquisition de ces séquences est effectuée durant une phase d'apprentissage où le robot est guidé par un opérateur humain. Ces séquences correspondent aux images que devraient obtenir la caméra embarquée durant la navigation autonome. L'opérateur délimite généralement les différentes séquences manuellement. Rasmussen [Rasmussen 96] propose un découpage automatique, en fonction de l'apparition ou disparition de primitives visuelles. L'environnement contient alors soit directement la réponse du capteur (l'image entière) [Matsumoto 96, Jones 97] soit simplement la position d'amers [Burschka 01, Gaspar 00]. Dans [Matsumoto 00a], le robot effectue automatiquement la saisie des séquences d'images. Cette méthode peut donc être associée au SLAM, bien que l'environnement ne soit pas reconstruit explicitement, et que sa représentation reste topologique. L'environnement de navigation considéré est une intersection de couloirs et est donc très structuré.

Pour faciliter les tâches de navigation à venir, certaines représentations stockent des labels avec les séquences d'images. Ces informations caractérisent le comportement de navigation que doit avoir le robot [Smith 02]. Elles sont définies soit pour une séquence complète [Gaspar 00, Vassallo 00], soit pour chaque image [Matsumoto 96, Jones 97]. Ces labellisations sont généralement effectuées par un opérateur humain lors de la phase d'apprentissage. Dans [Katsura 03], cette tâche est automatirisée, en utilisant les informations fournies par l'odométrie.

Le taux d'échantillonnage d'une séquence d'images évolue en fonction de la complexité du déplacement à réaliser [Rasmussen 96, Gaspar 00, Vassallo 00], et en fonction du schéma de navigation utilisé. Dans [Burschka 01, Argyros 01], les modélisations utilisées pour la navigation permettent de limiter fortement le nombre d'images nécessaire.

Dans la plupart des méthodes basées sur une représentation de l'environnement par des séquences d'images, la localisation consiste à chercher dans la séquence la ou les images les plus proches de celle fournie par la caméra. Le robot étant asservi sur la séquence d'images, celui-ci reste proche des positions associées. Des mesures de corrélation sont donc souvent utilisées pour réaliser la localisation. La corrélation considère soit l'image complète [Matsumoto 96, Inoue 93, Jones 97, Vassallo 00], soit des primitives d'intérêt comme des points [Rasmussen 96, Burschka 01]. [Argyros 01] utilise le suivi de primitives KLT.

Généralement, les méthodes de localisation proposées effectuent une localisation *intra-séquence*. Si l'environnement est décrit par plusieurs séquences, il est supposé que le robot connaît sa position initiale. Ces méthodes sont donc adaptées pour localiser le robot durant ses mouvements, par rapport à *une* séquence donnée. Mais elles ne permettent pas de sélectionner parmi plusieurs séquences celles où se situe initialement le robot. Si l'environnement est décrit par plusieurs séquences, cette information *a priori* est fournie par un opérateur.

Certaines représentations permettent cependant d'effectuer une localisation automatique, à la fois du robot et de la séquence associée. Dans [Gaspar 00], la localisation est indépendante des séquences d'images. La méthode utilisée, basée sur une projection de l'image par analyse de ses

composantes principales (ACP) est souvent utilisée dans les représentations par vues.

L'association d'une stratégie de navigation à un ensemble d'images revient finalement à regrouper dans le graphe les vues par rapport au comportement que doit y avoir le robot. De ce fait ces méthodologies sont très proches de celles qui consistent à regrouper explicitement les images par lieux, comme nous allons le voir dans la partie suivante. Une différence notable entre ces deux stratégies est que la représentation par séquence ne permet de localiser le robot que dans la séquence courante, alors que la représentation par lieux permet de déterminer celui dans lequel le robot se situe.

2.1.2.2 Représentation par lieux

Dans une telle représentation, les vues décrivant le même lieu sont regroupées dans une même entité. Ce regroupement est généralement effectué par un opérateur humain, comme dans [Ulrich 00] et [Zhou 03]. Košecká propose de regrouper automatiquement les vues [Košecká 03]. Pour ce faire une méthode de « clustering », le LVQ (pour *Learning Vector Quantification*), est appliquée sur des histogrammes des orientations des gradients extraits des images.

La localisation effectuée est purement qualitative : il est en effet recherché à quel lieu appartient l'image courante. Elle s'effectue généralement par recherche des plus proches voisins. Dans [Ulrich 00], les images sont décrites par des histogrammes de couleur TLS (teinte, luminance et saturation) et RVB. La mesure de similarité utilisée est la distance de Jeffrey, qui est très proche de $\mathcal{X}^2$. Ce descripteur est global, et ne prend pas en compte l'aspect géométrique de l'image. Deux images au contenu différent peuvent présenter les mêmes histogrammes et par conséquence faire échouer la localisation. C'est pourquoi [Zhou 03] propose comme mesure des histogrammes multi-dimensionnels prenant en compte les positions relatives des pixels dans les images. La mesure de similarité considérée est la divergence de Jeffrey [Puzicha 97]. Dans [Košecká 03], un lieu est décrit par un seul descripteur, prototype obtenu suite à la phase de clustering. Celui-ci est comparé à l'histogramme de l'image à localiser par la distance $\mathcal{X}^2$. La recherche des plus proches voisins permet d'obtenir une mesure de confiance, en comparant la réponse des deux lieux les plus proches.

2.1.2.3 Description par vues

Un environnement représenté par un ensemble de vues est décrit par un graphe topologique où chaque nœud est associé à une vue de la scène. La localisation du système robotique revient à chercher dans la base les images les plus proches en terme de contenu de la vue fournie par la caméra embarquée. Pour que ce processus réussisse, il est nécessaire de définir des descripteurs d'images suffisamment discriminants. L'analyse des composantes principales (ACP [Jolliffe 86]) du gradient du signal lumineux permet de diminuer considérablement la quantité d'information à stocker dans la base, tout en conservant les axes contenant le maximum d'information. [Kröse 99] utilise une telle description des images dans un cadre probabiliste. Les cinq premiers vecteurs propres sont utilisés pour décrire une image. La localisation s'effectue ensuite par une recherche des plus proches voisins pour chacun des vecteurs propres, en utilisant comme mesure de similarité la loi de Bayes. Dans [Paletta 01], les axes principaux sont aussi utilisés pour décrire des images panoramiques préalablement divisées en un certain nombre de segments. Dans un

cadre bayésien, et en tenant compte de la contrainte de continuité spatiale entre les segments, la localisation s'effectue aussi par recherche des plus proches voisins. Dans [Menegatti 04], les images panoramiques sont décrites par des transformées de Fourrier. Une localisation hiérarchique est réalisée en mesurant la dissimilarité entre deux vues avec plus ou moins de niveaux de décompositions de Fourier. Plus on utilise de niveaux de transformée, plus les vues devront être proches pour fournir un score de dissimilarité faible.

Plutôt que d'employer un descripteur de l'image entière, certains travaux proposent d'utiliser des primitives caractéristiques détectées dans l'image. Dans [Sim 99], les amers sont choisis comme les maxima locaux d'une fonction de densité associée à l'image. Chacun de ces amers est ensuite décrit par une analyse des composantes principales de l'intensité lumineuse dans son voisinage. Pour être robuste aux changements de point de vue, la description locale des différents amers est sauvegardée pour plusieurs positions de la caméra. Lors de la localisation, la distance euclidienne permet de sélectionner pour chaque amer de l'image à traiter son plus proche voisin dans la base. La position du robot est alors obtenue par une moyenne robuste des positions associées à chacun des amers sélectionnés dans la base. Il est nécessaire de sauvegarder plusieurs descriptions locales pour un même amer physique de la scène car la description utilisée n'est pas invariante aux transformations images.

Dans [Wolf 02], une description locale composée d'histogrammes multi-dimensionnels est obtenue par intégration d'un ensemble de fonctions sur l'espace des transformations euclidiennes. Cette intégration permet d'obtenir des mesures invariantes aux translations et aux rotations. Dans un premier temps, la localisation s'effectue par recherche des plus proches voisins, en utilisant comme mesure de similarité un opérateur d'intersection normalisé. Celle-ci est ensuite précisée par un filtrage bayésien (localisation de Markov).

Le lecteur attentif aura sans nul doute remarqué que les techniques de localisation employées dans le cas de représentation par vues et par lieux sont, si ce n'est identiques, tres proches. La différence entre ces deux approches réside dans la précision attendue de la localisation. Une localisation par lieux fournit le lieu dans lequel est supposé se trouver le système robotique, sans pour autant spécifier quelle vue de la base est la plus semblable à ce qu'observe actuellement le robot. Une localisation par vues va par contre donner cette information. La notion de lieux, en tant qu'information sémantique de plus haut-niveau, n'est dans ce cas pas recherchée. Enfin, si une information de type « lieu »apparaît satisfaisante pour un opérateur humain, elle n'est pas suffisamment précise et donc insuffisante pour permettre à un robot de contrôler ses mouvements.

2.1.3 Positionnement

Les méthodes basées sur les cartes de navigation et les décompositions en cellules se basent sur une hypothèse forte. En effet, elles supposent que la géométrie de la scène est connue de manière exacte, sans pour autant proposer, dans leur formalisme, une technique de reconstruction particulière. Les environnements étudiés sont clairement structurés. Ces méthodes sont donc plutôt adaptées à des applications en environnement intérieur, où les obstacles peuvent facilement être modélisés par des structure géométriques simples. Enfin, les exemples présentés dans les divers articles cités semblent indiquer que ces algorithmes s'intéressent particulièrement à la description d'une ou de quelques pièces. Tous ces éléments mènent à penser que ces méthodes sont difficiles à adapter pour des environnements extérieurs, et qui plus est, de taille conséquente.

Les méthodes assurant simultanément la localisation et la modélisation peuvent actuellement traiter des environnements extérieurs avec des capteurs de vision. Bien que de nombreux articles présentent des algorithmes temps réel, le passage à un environnement très vaste doit fatalement entraîner un coût de traitement supplémentaire potentiellement nuisible à une implantation à cadence vidéo. L'une des principales propriétés de ces approches est la prise en compte du bruit des mesures dans le schéma de modélisation, permettant de mieux localiser les primitives et de mieux localiser le robot.

De plus, les approches SLAM ne permettent pas exactement de répondre au problème que nous nous sommes initialement posé. Nous voulons pouvoir localiser un système robotique sans aucune information *a priori* sur la position qu'il peut initialement occuper, et ceci sans effectuer de mouvements. La grande majorité des méthodes de SLAM s'appuient sur les positions précédentes du robot pour le localiser au cours de son déplacement. Cette contrainte de localisation sans mouvement apparaît difficile à prendre en compte dans le cadre des méthodes de SLAM.

Enfin, posséder un modèle $3D$ précis d'un environnement est une hypothèse contraignante qui ne nous semble pas impérative pour effectuer une localisation du robot. Finalement, ces méthodes se focalisent sur le passage de l'information obtenue dans l'espace du capteur vers l'espace de travail du robot. Il est légitime de penser, les méthodes de représentation basée apparence nous le confirmant, que travailler directement dans l'espace du capteur, au moyen de descriptions adaptées, peut suffire à localiser correctement le robot.

La plupart des méthodes basées sur l'apparence nous semblent fortement contrôlées par les opérateurs humains. Les techniques stockant des séquences de trajectoires contraignent le robot à rejouer la trajectoire apprise. De plus, une modification de l'environnement peut rendre irréalisable la trajectoire apprise. La représentation de l'environnement n'est alors plus valide, et une phase d'apprentissage doit à nouveau être effectuée.

Les méthodes associant des labels aux prises de vue contraignent le robot à effectuer les mouvements « ordonnés » par l'opérateur sans lui laisser le choix de s'adapter. Si l'on considère qu'un système robotique connaît ses limites ou contraintes matérielles, il peut très bien lui-même se définir les trajectoires les plus faciles à réaliser. Nous voulons donc laisser cette tâche au système robotique.

Les méthodes représentant un environnement par un ensemble de vues ou de lieux regroupent généralement les informations par proximité spatiale. Si cette approche est adaptée pour une tâche particulière de localisation, elle nous semble inadaptée à la préparation d'une tâche de navigation, puisqu'elle ne permet pas de prendre en compte les contraintes de mouvement du robot. Nous estimons donc que la structure interne de la mémoire visuelle, même si celle-ci est constituée d'informations visuelles, ne doit pas définir des relations de proximité spatiale, mais plutôt des proximités en terme de mouvement. En effet, si la mémoire visuelle du robot doit lui permettre de « reconnaître » l'image de l'espace environnant que lui fournit sa caméra, elle doit aussi lui permettre de se « rappeler » comment il peut se rendre à la position désirée avec le moindre effort.

La section suivante présente comment, à partir de cette analyse, nous proposons de modéliser l'environnement de navigation, non seulement pour permettre à un système robotique de se localiser, mais aussi pour que ce dernier puisse se définir un chemin topologique vers la position désirée.

2.2 Une mémoire visuelle de l'environnement

Dans cette partie, nous décrivons la méthode que nous proposons pour représenter un environnement de navigation. Notre approche est basée apparence ; le capteur utilisé étant une caméra, l'espace de navigation est décrit par une collection d'images caractérisant l'environnement que peut observer la caméra durant ses déplacements. Ces prises de vue sont acquises durant une phase d'apprentissage, où le système robotique est déplacé par un opérateur humain. Dans notre cas, l'opérateur ne fournit au système aucune information sur la scène. Le traitement de la base est automatique. L'extraction d'information des images, et l'organisation de ces informations permettent non seulement de localiser le robot, mais aussi de définir l'environnement que doit observer la caméra durant la réalisation de la navigation.

La figure 2.1 présente succinctement les différentes opérations permettant au système robotique de traiter une tâche de navigation. La position initiale du robot avant le début du déplacement est décrite par l'image acquise par la caméra. De la même manière, la position qu'il doit atteindre est définie par la vue que doit fournir la caméra à la fin du déplacement.

Comme dans les méthodes basées sur une représentation par vues de l'environnement, la localisation du système robotique consiste à rechercher dans la base d'images celles qui sont les plus proches de l'image initiale en terme de contenu. De la même manière, la recherche des plus proches vues de l'image finale permet de localiser qualitativement la position désirée (sur le schéma 2.1, le terme « requête » désigne une image inconnue, initiale ou finale, qui doit être localisée). Cette étape est présentée dans la première section de cette partie.

Connaissant (qualitativement) sa position initiale et celle qu'il doit atteindre, le robot doit ensuite mettre en relation ces deux informations. Plus exactement, il doit définir l'environnement que sa caméra va observer durant la navigation. Cette opération revient à trouver un *chemin d'images* dans le graphe topologique qui est associé à la base d'images. La section 2.2.2 décrit la construction de ce graphe, et la section 2.2.3 présente la méthode de recherche de chemin d'images.

La section 2.2.4 résume les informations que doit contenir la représentation de l'environnement afin de permettre les opérations de localisation du robot et de détermination des prises de vue « utiles » pour définir visuellement l'environnement à traverser durant la navigation. Cette section s'attachera notamment à distinguer les traitements réalisés hors-ligne, lors de la création de la base, des opérations effectués en ligne, afin de résoudre une tâche de navigation.

2.2.1 Localisation basée apparence

Afin de bien nous situer par rapport aux méthodes existantes, il nous semble important de préciser que nous ne voulons pas obtenir une localisation précise ou absolue du système robotique. En effet, la localisation effectuée ici est qualitative ; nous recherchons les images de la base qui sont les plus proches de la prise de vue actuelle de la caméra. La localisation initiale est donc directement effectuée dans l'espace du capteur sans remonter dans un espace $3D$ lié à la scène, ou encore dans un espace associé à la pose du robot.

La localisation de l'image dite cible (celle que doit obtenir la caméra à la fin du déplacement) est effectuée de la même manière que celle de l'image initiale (acquise avant le déplacement). Ces deux images sont considérées comme des requêtes que nous voulons localiser par rapport aux vues formant la représentation de l'environnement.

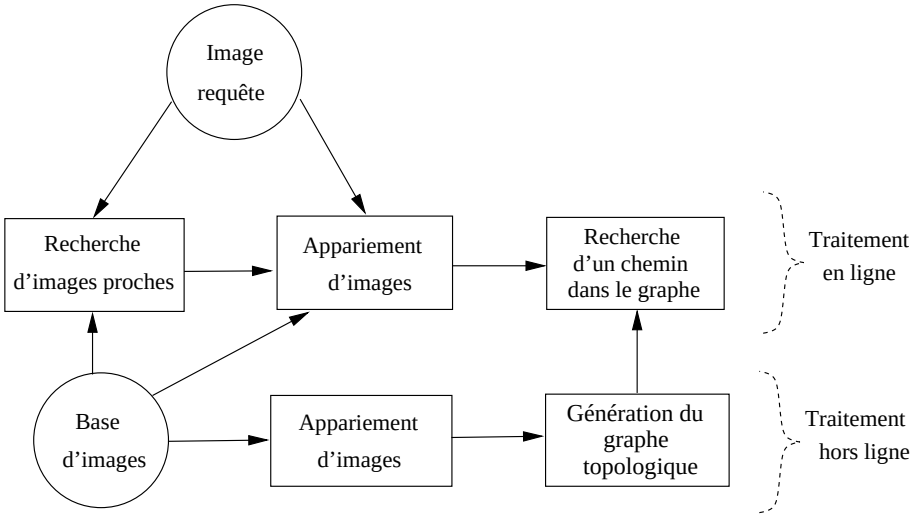


FIG. 2.1: Opérations réalisées pour définir une tâche de navigation (l'image requête est celle fournie par la caméra avant son déplacement).

Pour de nombreuses approches de navigation basées apparence, la localisation est traitée comme un problème de reconnaissance d'images. La partie 2.1.2 en a présenté certaines. La section suivante montre explicitement la similarité de ces deux problématiques.

2.2.1.1 La reconnaissance d'images comme outil de localisation

Le domaine de la reconnaissance d'images est actuellement très actif, de part la prolifération des images numériques [Smeulders 00, Veltkamp 00]. Elle est basée sur la définition d'un ou plusieurs descripteurs permettant de caractériser le contenu d'une image.

Certaines méthodes de reconnaissance utilisent des descripteurs globaux, calculés sur l'image entière. C'est le cas par exemple des histogrammes de couleur, de niveaux de gris ou encore des corrélogrammes [Stricker 94, Huang 97]. Si ces descripteurs sont peu sensibles au bruit du signal, ils ne préservent aucune géométrie et s'avèrent peu discriminants. Par ailleurs, le calcul de la similarité entre histogrammes s'avère coûteuse, les distances simples étant très peu satisfaisantes. Enfin ce type de descripteur est très sensible aux traitements de recadrage des images, et n'est pas adapté à la détection des parties communes entre deux vues.

En robotique, de nombreux travaux utilisent de tels descripteurs pour réaliser la localisation. Les histogrammes de couleur sont par exemple utilisés dans [Ulrich 00, Yuen 02, Zhou 03]. Dans le cas de la localisation en environnement intérieur, la prédominance des amers rectilignes (contours du mobilier, encadrement des portes, extrémité d'un couloir, ...) permet l'utilisation d'histogrammes basés sur les orientations du gradient de l'image [Košecká 03]. D'autres travaux extraient directement de ce gradient un descripteur de taille moindre constitué de ses caractéristiques prédominantes, par analyse des composantes principales [Jogan 99, Kröse 99, DeLaTorre 01] (l'emploi du gradient permet d'obtenir une mesure invariante aux changements globaux d'illumination).

La sensibilité des descripteurs globaux, problème avéré en reconnaissance d'images, peut de la même manière pénaliser la qualité de la localisation. En effet, l'environnement peut avoir évolué

depuis la phase d'acquisition de la base d'images le décrivant. L'image qui doit être reconnue peut donc contenir des objets non « décrits » dans la base. De plus, certains objets observés lors de l'apprentissage peuvent ne plus être présents dans la scène. Ces évolutions imposent d'utiliser une technique de localisation robuste à ces perturbations locales. L'utilisation de descripteurs globaux ne permet pas de respecter cette contrainte.

L'utilisation de caméras panoramiques permet de minimiser la part de bruit apporté par ces modifications de l'environnement [Jogan 99, Gaspar 00, Paletta 01]. Ce type de caméra permet donc l'utilisation des descripteurs globaux pour la localisation. Cependant, si le système ne possède qu'une caméra classique, les éventuelles modifications locales de l'environnement rendent inefficaces ces descripteurs.

Une autre approche en reconnaissance d'images consiste à associer à une prise de vue un ensemble de descripteurs locaux [Schmid 97, Lowe 04]. Chacun de ces descripteurs décrit une petite partie ou région de l'image. La mise en relation de deux prises de vue repose alors sur le nombre de descripteurs qui sont proches selon une métrique choisie. L'aspect local de la description permet de conserver et caractériser les détails constituant une image. La reconnaissance est alors plus précise, et reste performante si les images subissent des recadrages. La reconnaissance d'images s'effectue en recherchant pour chaque descripteur requête ceux de la base qui sont les plus proches. En « sommant » ces résultats obtenus localement, les images de la base sont classées en fonction du nombre de leurs descripteurs qui ont été mis en relation avec ceux de l'image requête.

Ce type de recherche d'images apparaît bien adapté à la localisation basée vision de systèmes robotiques. En effet, l'apparition ou la disparition d'objets dans la scène depuis la phase d'apprentissage reste une perturbation locale de l'image requête. Le contenu du reste de l'image est par contre inchangé, par rapport à la connaissance de l'environnement qu'en a le système. Les descripteurs locaux de ces zones suffisent alors pour effectuer la reconnaissance de l'image requête. De plus, certains descripteurs locaux sont robustes aux différentes transformations que peuvent subir les images (rotation, translation, changement d'illumination). Il est donc possible de mettre en relation deux images décrivant une même scène depuis deux points de vue différents.

La technique que nous proposons d'utiliser rentre dans cette deuxième famille. Les descripteurs locaux correspondent à une description photométrique du signal en niveau de gris dans le voisinage d'un point d'intérêt [Schmid 97]. Nous détaillons ci-dessous comment ces descripteurs locaux sont définis. Nous reviendrons ensuite sur la méthode de reconnaissance d'images à partir de tels descripteurs.

2.2.1.2 Caractérisation locale d'une prise de vue

Les descripteurs locaux employés appartiennent à la famille des invariants différentiels locaux [Florack 94]. Leurs définitions s'effectuent en trois étapes : la détection de points d'intérêt, la caractérisation du signal dans le voisinage de ces points par le calcul des dérivées du signal, et enfin la combinaison de ces dérivées pour obtenir des descripteurs aux propriétés d'invariance souhaitées.

Les points d'intérêt sont détectés suivant la méthode de Harris (voir la section 1.2.2 à la page 23). Schmid propose dans [Schmid 00] une amélioration du détecteur de base, afin de rendre plus stable le calcul des dérivées du signal. Pour cela, ces dérivées sont convoluées avec une gaussienne. Le détecteur résultant est alors très similaire au KLT.

Le signal est ensuite caractérisé dans le voisinage de chacun de ces points, en le convoluant avec une gaussienne, et en calculant les neuf premières dérivées jusqu'à l'ordre 3 (le voisinage choisi correspond au support des dérivées). On obtient pour chaque point d'intérêt $I(x, y)$ les grandeurs : $I, I_x, I_y, I_{xx}, I_{yy}, I_{xy}, I_{xxx}, I_{xxy}, I_{xyy}, I_{yyy}$.

L'invariance aux transformations images est alors obtenue par la combinaison de ces dérivées. Notons que l'invariance à la translation est assurée par le choix d'un système de coordonnées centré sur le point détecté, et par le fait que le détecteur Harris est indépendant des translations.

Pour obtenir l'invariance aux rotations, les supports sur lesquels sont calculés les dérivées sont choisis circulaires. Les dérivées sont ensuite combinées afin d'éliminer de manière algébrique l'angle de rotation. En notation d'Einstein, les neufs invariants obtenus sont :

$$
\begin{gathered}
I \\
I_i I_i \\
I_i I_{ij} I_j \\
I_{ii} \\
I_{ij} I_{ji} \\
\epsilon_{ij} \left(I_{jkl} I_i I_k I_l - I_{jkk} I_i I_l I_l \right) \\
I_{iij} I_j I_k I_k - I_{ijk} I_i I_j I_k \\
-\epsilon_{ij} I_{jkl} I_i I_k I_l \\
I_{ijk} I_i I_j I_k
\end{gathered}
$$

où ϵ_{ij} est défini par $\epsilon_{xy} = -\epsilon_{yx} = 1, \epsilon_{xx} = \epsilon_{yy} = 0$. La notation d'Einstein correspond à la sommation sur les indices répétés. Par exemple, $I_i = \sum_i I_i = I_x + I_y$, et $I_{ij} I_{ji} = \sum_i \sum_j I_{ij} I_{ji} = I_{xx} I_{xx} + 2 I_{xy} I_{xy} + I_{yy} I_{yy}$.

Les changements d'illumination sont modélisés par un modèle affine : $I' = aI + b$. Pour obtenir des grandeurs invariantes à de telles transformations, il suffit de considérer les sept derniers invariants, et de les diviser par la puissance adéquate de $I_i I_j$, afin d'éliminer le facteur a. Le vecteur ainsi obtenu correspond au descripteur d'un point d'intérêt.

Notons que des extensions ont été proposées afin de pouvoir définir des invariants photométriques avec des images couleurs [Amsaleg 01]. Dans notre cas, nous nous limitons aux images en niveaux de gris.

La section suivante présente comment s'effectue la reconnaissance d'une image requête en utilisant ces invariants photométriques.

2.2.1.3 Indexation d'images sur les invariants photométriques

Tout d'abord, toutes les images de la base sont traitées pour en extraire ces descripteurs locaux. Lorsqu'une image requête est présentée, celle-ci est de la même manière analysée pour lui associer un ensemble de descripteurs requêtes. Chacun de ces derniers sont ensuite mis en relation avec la base, par une recherche de leurs k-plus proches voisins. Les images auxquelles sont associées les descripteurs proches se voient alors attribuées un vote. Une fois tous les descripteurs traités, les images de la base sont classées en fonction du nombre de votes qu'elles ont obtenu. Les vues ayant le plus de voix sont alors considérées comme les images les plus proches de l'image requête.

Le test de Hinkley permet de choisir les images pertinentes et de ne pas en rester à un simple classement [Berrani 04]. Pour ce faire, la liste ordonnée des scores est assimilée à une fonction. Le test de Hinkley détecte alors le seuil de transition, caractérisé par une forte variation des scores.

Dans [Berrani 03], une optimisation du temps de recherche a été effectuée en regroupant les descripteurs de la base dans des « clusters », et en effectuant une recherche approximative. Une telle optimisation serait utile pour une application grandeur nature. Dans le cadre de nos travaux prospectifs, nous nous sommes limités à une implémentation séquentielle.

2.2.1.4 Résultats expérimentaux

Nous présentons ici quelques résultats de recherche d'images basée sur le contenu. L'algorithme de recherche que nous avons utilisé a été développé dans le cadre de la thèse de Sid-Ahmed Berrani [Berrani 04].

La figure 2.2 présente une recherche pour le cas d'un environnement plan de navigation. Lors de la phase d'apprentissage, une cinquantaine de prises de vue ont été acquises en déplaçant la caméra sur un plan parallèle au plan de la scène. L'image (a) correspond à l'image requête. Elle n'est pas présente dans la base. Les images de la base sont décrites par en moyenne 150 points d'intérêt. Les images suivantes sont les images les plus proches trouvées dans la base (pour une recherche des 15 plus proches voisins).

La recherche d'images est facilitée puisque la scène est plane. En effet, le descripteur caractérisant un point dans différentes images reste quasiment identique, aucune variation de profondeur relative ne venant perturber son extraction. Notons tout de même que les conditions d'illumination entre l'image requête et les images de la base ne sont pas identiques, et que les résultats de recherche n'en pâtissent aucunement.

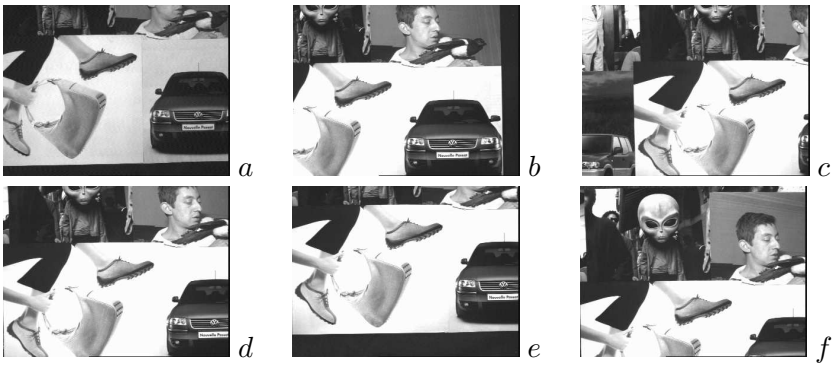


FIG. 2.2: Résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 57$, $c = 54$, $d = 52$, $e = 50$, $f = 47$

Les expériences suivantes ont été réalisées sur des données obtenues dans le cadre du projet ROBEA BODEGA. Les bases d'images utilisées proviennent de deux séquences d'images acquises par un système stéréoscopique embarqué dans une voiture. La table 2.3 décrit le contenu des différentes bases que nous avons générées à partir de ces données. La figure 2.4 illustre la grande variété des environnements traversés durant l'acquisition de ces séquences.

Dans toutes ces expériences l'image requête n'est pas contenue dans la base. Sur les figures 2.5 et 2.7, les prises de vue recherchées correspondent à des vues acquises avec la caméra de gauche. Les nombres de votes (ou scores) obtenus par les images dépendent du contenu de la base, mais aussi du nombre de descripteurs calculés dans l'image requête. De manière générale, le nombre des vues possédant au moins une voix est très grand. En effet, plus la taille de la base d'images

augmente, plus il y a de chances de trouver pour un descripteur requête des descripteurs de la base qui sont proches au sens de la mesure de similarité, mais qui ne correspondent pas au même point physique $3D$. Par exemple, dans le cas de la première recherche (figure 2.5), environ 73% des images de la base ont obtenu au moins une voix. Cependant, seulement 2% de ces prises de vue ont obtenu un score supérieur à 5.

La large dispersion des voix peut sans doute justifier le fait que le classement obtenu ne nous apparait pas toujours, visuellement parlant, comme étant le plus cohérant. Ainsi, sur la première recherche, on aurait tendance à mieux classer la vue (e) que la vue (b). Cependant, du point de vue du signal, il est tout à fait possible que les distances mesurées entres les descripteurs des vues (b) et (a) soient plus faibles que celles mesurées entre (e) et (a). Ce fait justifie ainsi l'intérêt de ne pas se limiter à la recherche de la vue *la plus proche* mais au contraire d'effectuer la recherche *des k-plus proches* images de la base.

Nom	Caractéristiques	Nombre d'images	Nombre de descripteurs
Base 1	Séquence 1, caméra de gauche	1536	316 416
Base 2	Séquence 1, caméra de gauche (1 vue sur 3) et caméra de droite (1 vue sur 3)	684	137 484
Base 3	Séquence 2, caméra de gauche	1255	533 375
Base 4	Séquence 2, caméra de gauche (1 sur 3) et caméra de droite (1 sur 3)	838	345 256
Base 5	Base 2 et Base 4	1522	482 740

FIG. 2.3: Caractéristiques des bases d'images de test

Dans les expériences 2.6 et 2.8, la base est constituée d'images des deux caméras du système stéréoscopique. Dans les deux cas, l'image requête correspond à une prise de vue de la caméra de droite. Les prises de vue les plus proches trouvées correspondent à des prises de vue des deux caméras. Cependant, nous avons constaté que les images les mieux classées sont des images qui ont été acquises avec la même caméra que celle de l'image requête. Sur la figure 2.6 (respectivement 2.8), les images b, c, e (resp. a, b, c, d) ont été prises par la caméra de droite. Cette particularité nous semble due au fait que le signal lumineux caractérisant une scène tridimensionnelle varie moins lors d'un déplacement le long de l'axe optique que lors d'un déplacement latéral.

Les deux derniers exemples (figures 2.9 et 2.10) illustrent l'efficacité de l'algorithme de localisation lorsque l'environnement est constitué de la description de plusieurs chemins. Un ensemble de vues des deux séquences 1 et 2 constitue ici la base d'images de l'environnement. L'image requête de la figure 2.9 est une vue de la première séquence, alors que celle de 2.10 provient de la deuxième séquence. Dans les deux cas encore, les images les plus proches proviennent de la même caméra du système stéréoscopique que celle qui a prise l'image requête (a, b, c, d, e pour la première figure, a, b, c, d, g pour la deuxième).

FIG. 2.4: Illustration de la variété des images de la base (Les quatre premières vues concernent la séquence 1, et les autres la séquence 2)

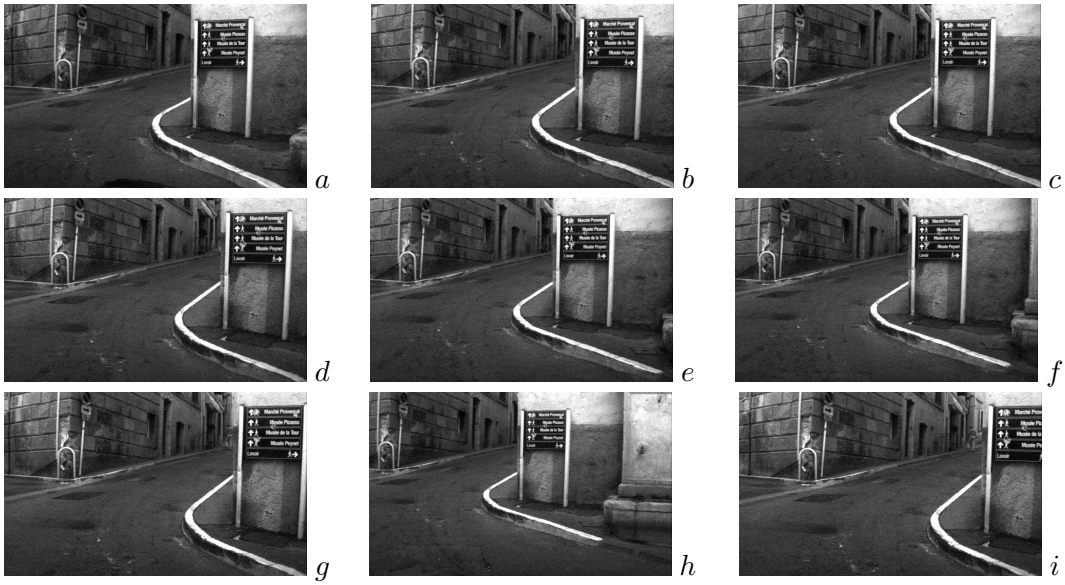


FIG. 2.5: Base 1 : résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 46$, $c = 42$, $d = 38$, $e = 35$, $f = 25$, $g = 18$, $h = 13$, $i = 10$

FIG. 2.6: Base 2 : résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 73$, $c = 68$, $d = 45$, $e = 42$, $f = 40$, $g = 38$, $h = 33$, $i = 31$

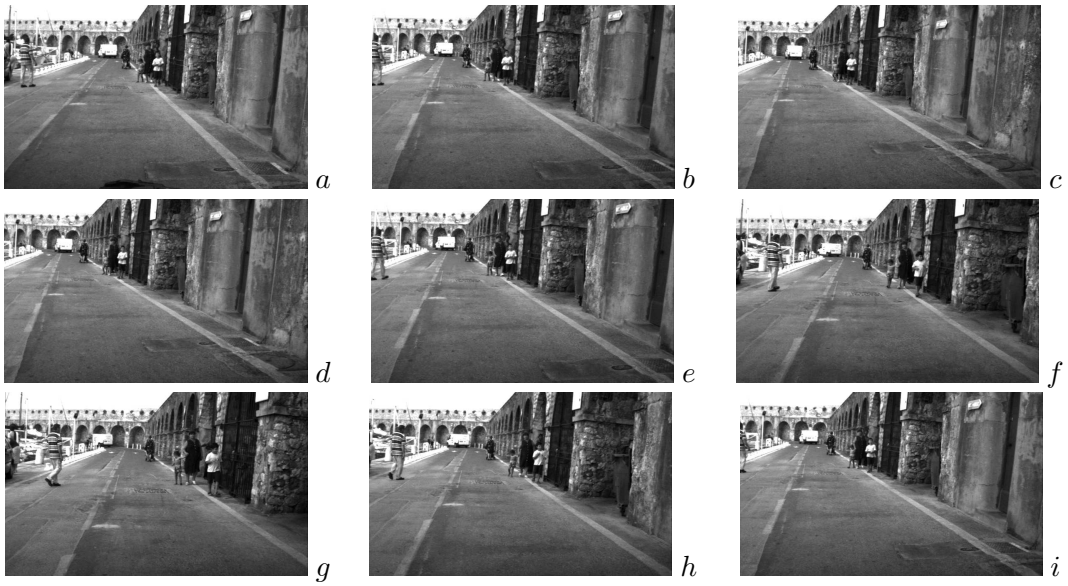


FIG. 2.7: Base 3 : résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 27$, $c = 21$, $d = 19$, $e = 18$, $f = 18$, $g = 18$, $h = 17$, $i = 16$

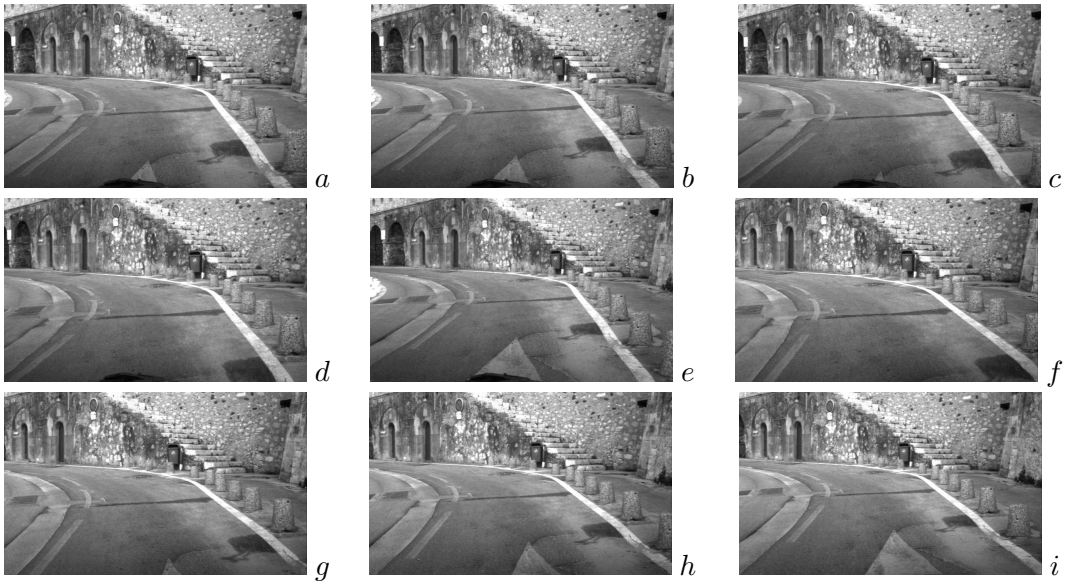


FIG. 2.8: Base 4 : résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 75$, $c = 66$, $d = 56$, $e = 52$, $f = 52$, $g = 47$, $h = 46$, $i = 45$

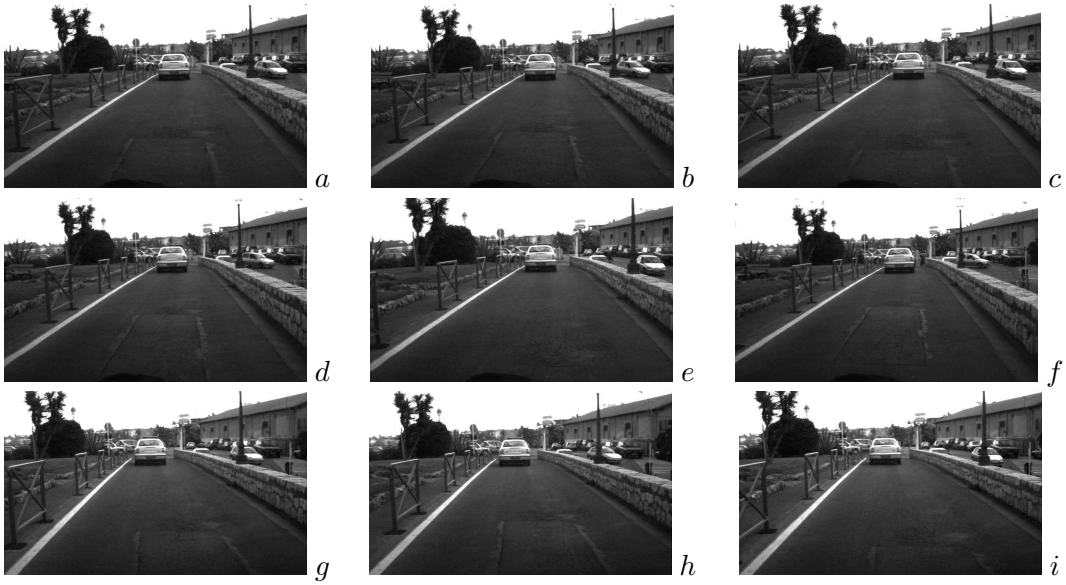


FIG. 2.9: Base 5 : résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 35$, $c = 25$, $d = 23$, $e = 18$, $f = 16$, $g = 15$, $h = 13$, $i = 11$

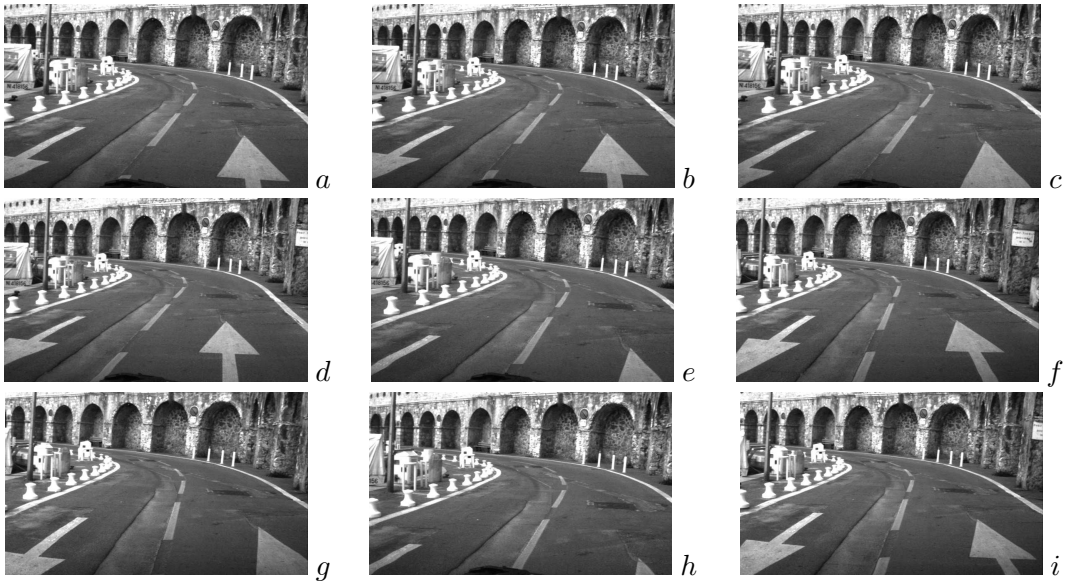


FIG. 2.10: Base 5 : résultats de l'indexation pour l'image requête a. Les scores obtenus sont : $b = 110$, $c = 92$, $d = 81$, $e = 73$, $f = 69$, $g = 69$, $h = 67$, $i = 66$

Ces expériences démontrent ainsi que la « localisation » qualitative des images initiale et finale peut s'effectuer en utilisant les principes de la recherche d'images. L'approche proposée permet donc bien de répondre aux deux premières questions « Où suis-je ? » et « Où vais-je ? ». La partie suivante présente comment ces deux prises de vue sont mises en relation par l'extraction d'un chemin d'images, grâce à une organisation adaptée de la mémoire visuelle du système robotique.

2.2.2 Structuration de la mémoire visuelle de l'environnement

Pour pouvoir répondre à la question « Que dois-je voir durant mon déplacement ? », il est nécessaire de définir et mettre en place une organisation des images de la base. Comme toutes les techniques de représentation basée sur l'apparence, cette organisation passe par la définition d'un graphe topologique associé à la base.

Dans la littérature, la plupart des graphes topologiques associant un nœud à une prise de vue ne propose pas au système robotique plusieurs chemins pour se déplacer d'un lieu vers un autre (la figure 2.11(a) illustre ces propos). Un noeud est généralement connecté au noeud correspondant à la position d'où est censé arriver le robot, et au noeud où est censé se rendre ensuite le robot (nous ne considérons pas ici les configurations associées à un croisement de différentes routes, où naturellement les connexions entre nœuds sont plus importantes). Cette caractéristique repose sur l'hypothèse que durant la phase d'apprentissage, le robot n'a été déplacé qu'une seule fois dans la zone considérée.

Cette hypothèse nous apparaît cependant restrictive. Déplacer plusieurs fois le système robotique dans une même zone permettrait d'obtenir une meilleure description de l'environnement, et de choisir les informations qui apparaissent, pour une position donnée, les plus pertinentes dans les différentes séquences. De plus, ce *modus operandi* permettrait de filtrer le bruit lié à la « maudite » propriété dynamique (pour les robots) des environnements de navigation. Par exemple, durant la

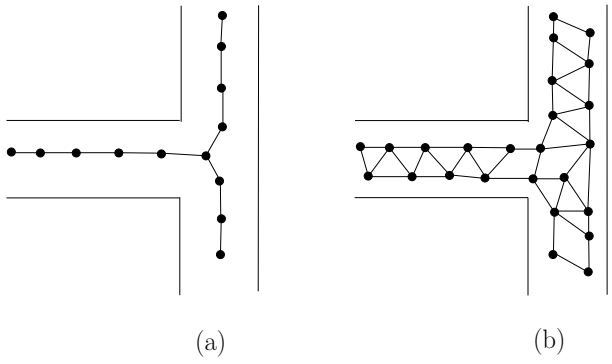


FIG. 2.11: Graphe topologique (a) : approche classique, (b) : approche considérée

phase d'apprentissage, un piéton traverse la route. Celui-ci sera donc présent dans l'image acquise par la caméra, alors qu'il ne constitue clairement pas un élément caractéristique de l'environnement. Ou encore le système robotique est amené à tourner pour éviter un piéton ou un véhicule se situant dans sa voie de navigation. Cet évitement d'obstacle lors de l'apprentissage n'a pas lieu d'être dans la représentation de l'environnement.

En supposant que durant l'apprentissage plusieurs passages peuvent être réalisés dans une même zone, la structure du graphe devient alors proche de celle présentée sur la figure 2.11(b). Un noeud peut donc être relié à plus de deux nœuds.

Cette structure donne ainsi la possibilité au système robotique de choisir la succession de nœuds (ou d'images) qui lui semble le plus adapté à la définition de l'environnement qu'il doit observer durant la navigation. Si ce chemin doit impérativement former une description continue de cet environnement, il peut aussi être intéressant qu'il corresponde implicitement à un ensemble de mouvements faciles à réaliser. Si le système robotique peut aisément se déplacer dans toutes les dimensions de l'espace de travail, l'importance de la zone de recouvrement entre deux prises de vue permet de définir la facilité du mouvement. En effet, le recouvrement entre deux images est d'autant plus grand que les positions associées du système sont proches.

Cependant, pour des systèmes de type automobile, tous les mouvements n'ont pas le même coût. Typiquement, les déplacements latéraux sont beaucoup plus chers à réaliser que les déplacements en ligne droite.

C'est pourquoi nous proposons de prendre en compte, dès la construction du graphe topologique, la notion de facilité des déplacements. La première section présente deux manières différentes de chiffrer la difficulté d'un mouvement, en fonction du type de système robotique considéré.

La deuxième section s'intéresse au problème de la dimension de l'environnement de navigation. Plus cet environnement est grand, plus le graphe associé est important. La recherche de chemins dans le graphe peut alors devenir très coûteuse en temps de calcul. Pourtant, le chemin recherché ne nécessite pas obligatoirement de considérer l'ensemble du graphe. Les graphes hiérarchiques que nous proposons ont donc pour objectif de diminuer la taille du graphe à considérer lors d'une recherche de chemin.

Mais avant de rentrer dans le vif du sujet, un petit rappel sur les termes et notations dédiés à l'algorithmique des graphes s'impose [Froidevaux 92].

Définition 1 : Un *graphe simple et non orienté* $G = (X, A)$ est un couple formé de deux ensembles, où :
- les éléments de $X = x_1, x_2, \ldots, x_n$ sont appelés des *sommets* ou *nœuds*. Si l'on note $N = \text{card}(X)$ le nombre de nœuds du graphe, on dit que G est *d'ordre* N ;
- les éléments de $A = a_1, a_2, \ldots, a_m$ sont des *arcs*, et correspondent à des couples de sommets.

Définition 2 : Un *chemin*[1] Γ de longueur q d'un graphe simple et non orienté est une séquence de q nœuds :

$$\Gamma = x_1, x_2, \ldots, x_q,$$

où deux nœuds successifs x_i et x_{i+1} sont connexes (c'est-à-dire reliés par un arc).

Définition 3 : Un *graphe valué* est un graphe simple et non orienté $G = (X, A)$, muni d'une fonction $\gamma : A \rightarrow \mathbb{R}_+^*$, appelée *fonction de coût*, et qui associe un coût à chaque arc du graphe.

Définition 4 : Le *coût d'un chemin* est la somme des coûts des arcs de ce chemin.

2.2.2.1 Fonction de coût inter-image

La définition d'un coût pour chacun des arcs du graphe passe par l'analyse de chaque couple d'images de la base.

Une étape préliminaire consiste donc à détecter un ensemble d'amers des images de la base, et à les mettre en relation avec ceux définis dans les autres vues. Pour ce faire, nous utilisons l'algorithme d'appariement robuste de Zhang [Zhang 94] (la section 1.2.3 page 26 en donne une brève description). À chaque couple d'images de la base est alors associé un ensemble de primitives en correspondance (cet ensemble peut bien sûr être vide si les prises de vue ne décrivent pas une même zone de l'environnement).

Un arc est défini dans le graphe entre chaque couple d'images possédant des amers en commun. Clairement, si deux images contiennent des projections de mêmes primitives de la scène, elles correspondent à des positions proches du robot. Dans notre cas, nous choisissons un coût inversement proportionnel à la facilité du déplacement entre les deux images associées ; deux images n'ayant pas de correspondance sont reliées par un arc de coût infini positif. La recherche d'un chemin par l'algorithme de Dijkstra [Froidevaux 92] permettra ensuite d'obtenir un chemin de coût minimum entre deux nœuds du graphe.

Deux fonctions de coût différentes sont proposées pour valuer les arcs du graphe. L'une est directement basée sur les mises en correspondance, l'autre est dépendante du mouvement entre les images.

[1] le terme normalement utilisé est *chaîne*. Nous utilisons plutôt *chemin* au vu de la signification que nous donnons aux arcs du graphe.

Fonction de coût basée vision

Cette première fonction de coût est directement définie à partir du nombre de primitives en correspondance entre les différents couples d'images de la base. Cette fonction repose sur l'hypothèse que plus les amers appariés entre deux vues sont nombreux, plus le déplacement associé est facile. La fonction de coût γ est donc choisie comme l'inverse du nombre de primitives en correspondance. Si l'on note $n_{i,j}$ le nombre de primitives appariées entre les vues ψ_i et ψ_j du graphe, le coût de l'arc entre les deux nœuds x_i et x_j est :

$$\gamma(x_i, x_j) = \begin{cases} +\infty & \text{si } n_{i,j} = 0, \\ \frac{1}{n_{i,j}} & \text{sinon} \end{cases}$$

Cette fonction est satisfaisante si l'on considère un système robotique pouvant aisément se déplacer selon tous ses axes de liberté. Nous proposons une deuxième fonction de coût plus adaptée aux systèmes robotiques n'ayant pas autant de liberté de mouvement.

Fonction de coût basée mouvement

Si l'on considère une application en robotique mobile, et plus particulièrement pour des véhicules non holonomes[2], une valuation simplement basée sur les correspondances n'est pas efficace. En effet si l'on considère deux images prises suite à un déplacement latéral, de nombreuses primitives peuvent être mises en correspondance. Les deux images vont donc être considérées comme proches, ce qui n'est pas souhaitable puisqu'un tel déplacement est difficilement réalisable par un véhicule non holonome. Si l'on désire une fonction de coût traduisant efficacement la facilité qu'aurait le robot à se déplacer d'une position à l'autre, il est alors nécessaire de considérer le mouvement réalisé entre ces deux positions.

Nous avons conservé l'orientation du repère caméra utilisée en robotique pour représenter la pose d'une caméra (voir la figure 2.12). Clairement un véhicule n'a pas la même mobilité qu'un robot manipulateur. Ainsi, les mouvements le long de l'axe $\vec{y}$ lui sont impossibles. De la même manière, les seuls mouvements de rotation qu'il peut réaliser s'effectuent autour de l'axe $\vec{y}$. Notons cependant que les mouvements supposés interdits peuvent tout de même être mesurés entre deux images ; une route en pente entraîne une variation de la position suivant $\vec{y}$ du robot, un léger tangage est observé lorsque le véhicule tourne, ce qui correspond à une rotation sur $\vec{z}$, ... Ces mouvements peuvent être associés à la topologie de la route, ou à la dynamique du véhicule, et donc être ignorés lors de la détermination de la fonction de coût.

Finalement, lors d'un déplacement, nous voulons :
- favoriser les déplacements le long de l'axe optique ;
- éviter les déplacements latéraux ;
- éviter les rotations du véhicule.

Notons de plus qu'une rotation (ou un déplacement latéral) est plus aisée si elle est accompagnée d'un déplacement le long de l'axe optique.

[2] un robot non holonome comme une voiture, peut se déplacer vers l'avant, et l'arrière, tourner à gauche et à droite. Mais il ne peut pas se déplacer latéralement et tourner librement. Il possède moins d'entrées de commande (2 pour la voiture) que de degrés de liberté (3).

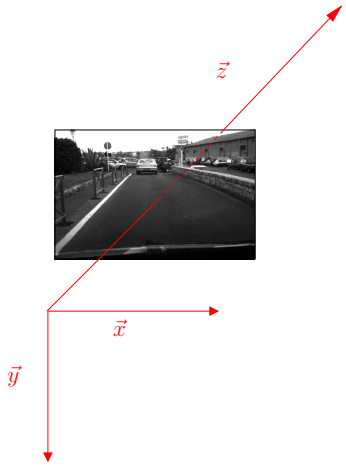


FIG. 2.12: Position et orientation utilisée pour le repère caméra

Comme il a été présenté dans le chapitre 1, le mouvement partiel $\left({}^{i}\mathbf{R}_j, \alpha {}^{i}\mathbf{t}_j\right)$ entre deux images ψ_i et ψ_j peut être estimé à partir de la décomposition[3] de l'homographie ${}^{i}\mathbf{H}_{n_j}$ associée à ces deux prises de vue. Dans le reste de ce chapitre, nous noterons ce mouvement partiel $\mathbf{R}$ et $\mathbf{b} = (b_x, b_y, b_z) = \alpha \mathbf{t}$.

La fonction de coût γ est alors définie comme étant :

$$\gamma(x_i, x_j) = \begin{cases} \infty & \text{si} \quad n_{i,j} = 0, \\ \gamma_\infty & \text{si} \quad b_z = 0, \\ 1 + \frac{|b_x| + \kappa\theta|u_y|}{|b_z|} & \text{sinon} \end{cases} \tag{2.1}$$

γ_∞ est une constante maximale associée aux mouvements purement latéraux (notons qu'en pratique, b_z ne vaut quasiment jamais exactement 0). $\mathbf{u}\theta$ est la représentation vectorielle de la rotation $\mathbf{R}$, avec $\mathbf{u} = (u_x, u_y, u_z)$ l'axe de rotation, et θ l'angle de rotation autour de cet axe. Une telle définition permet de fixer un coût minimum à chaque déplacement du système robotique. Si l'on considère que $\kappa = \alpha\eta$, et dans le cas où $b_z \neq 0$, la valuation devient :

$$\gamma(x_i, x_j) = 1 + \frac{|b_x| + \kappa\theta|u_y|}{|b_z|} = 1 + \frac{\alpha|t_x| + \alpha\eta\theta|u_y|}{\alpha|t_z|} = 1 + \frac{|t_x| + \eta\theta|u_y|}{|t_z|}.$$

Finalement, le terme $\kappa > 0$ a trois objectifs :

- il permet de prendre en considération la différence d'amplitude entre les translations et rotations ;
- il donne la possibilité de définir une préférence entre les rotations sur $\vec{y}$ et les translations sur $\vec{x}$;
- il permet de compenser le facteur d'échelle inconnu α.

[3] N'ayant aucune information *a priori* sur le plan de référence qui peut être choisi, l'ambiguïté sur la décomposition de ${}^{i}\mathbf{H}_{n_j}$ peut être levée en estimant deux homographies relatives à deux plans de références différents, et en gardant la décomposition commune.

Cette fonction de coût permet de définir un graphe valué associé à la base d'images. La section suivante présente comment un ensemble de graphes hiérarchiques peut être défini à partir de ce graphe initial, afin de diminuer par la suite l'espace de recherche de chemins.

2.2.2.2 Génération de graphes hiérarchiques

La complexité d'une recherche de plus court chemin dépend de la taille du graphe considéré. Si l'on considère un graphe d'ordre N composé de M arcs, cette complexité est de $\mathcal{O}(M + N\log(N))$. Si l'on considère un environnement vaste, la taille du graphe devient très importante, et la recherche d'autant plus longue à effectuer.

Pour accélérer la phase de recherche, nous proposons de diminuer la taille du graphe par la génération de graphes hiérarchiques. À partir du graphe initial $G_0(X_0, A_0)$ est défini un ensemble m de graphes $G_i(X_i, A_i)$ tels que :

- $|X_0| > |X_1| > \ldots |X_{m-1}| > |X_m|$
- $|A_0| > |A_1| > \ldots |A_{m-1}| > |A_m|$

Pour obtenir ces différents graphes, nous nous sommes inspirés des travaux sur le partitionnement de graphes [Hendrickson 93, Karypis 98]. Ces techniques sont généralement employées pour paralléliser des programmes. La génération des graphes de niveau supérieur s'effectue en deux étapes :

- la phase de *mise en relation* (ou *matching*), qui consiste à regrouper les nœuds considérés les plus proches,
- la phase de *regroupement* (*contracting*), qui permet de générer un graphe de niveau supérieur en regroupant les nœuds les plus proches dans une même entité.

Détection des nœuds proches

Une *mise en relation* est définie par un sous-ensemble des arêtes de G_0, noté $A_m \subseteq A$, tel qu'aucun couple de deux arêtes de A_m ne partage un même sommet. Soit $a_{i,j}$ l'arête entre les nœuds i et j. Si $a_{i,j} \in A_m$, alors aucune des arêtes de la forme $a_{i,k}$ et $a_{j,k}$, pour tout $k < m$, n'appartient à A_m.

Une *mise en relation maximale* est telle qu'aucun arc de A ne peut plus être rajouté à A_m. Différentes techniques de mise en relation ont été proposées dans la littérature [Hendrickson 93]. La technique employée ici est dérivée de la méthode du *Light Edge Matching* [Karypis 98], où les nœuds connectés par un arc de valeur minimal sont regroupés.

L'algorithme utilisé est présenté dans le tableau 1. Les mises en relation effectuées sont stockées dans une table notée *proche* (proche$[i]$ désignant le nœud mis en relation avec le sommet x_i). Deux nœuds x_i et x_j sont mis en relation si et seulement si x_i est le plus proche nœud de x_j **et** x_j est le plus proche nœud de x_i. De ce fait, des nœuds peuvent ne pas être associés à un autre nœud. La fonction Min(x_i) renvoie le nœud x_j avec $(x_i, x_j) \in A$, tel que la valuation $\gamma(x_i, x_j)$ est la plus faible :

$$\text{Min}(x_i) = (x_j | ((x_i, x_j) \in A) \wedge (\forall x_k \neq x_j (x_i, x_k) \geq (x_i, x_j)))$$

[Initialisation :]
Pour x_i **de** 1 **à** N **faire**
  proche$[x_i] =$ INDÉFINI ;
Fin Pour
[Boucle de traitement :]
Pour x_i **de** 1 **à** N **faire**
  $x_j = \text{Min}(x_i)$
  Si ((proche$[x_j] ==$ INDÉFINI) **et** $(\text{Min}(x_j) == x_i)$**) Alors**
    proche$[x_i] = x_j$;
    proche$[x_j] = x_i$;
  Sinon
    proche$[x_i] = x_i$;
  Fin Si
Fin Pour

Table 1: Algorithme de mise en relation maximale des nœuds

La figure 2.13 présente un exemple de mise en relation maximale. Aucun autre arc ne peut être rajouté à l'ensemble des arcs représentés en rouge.

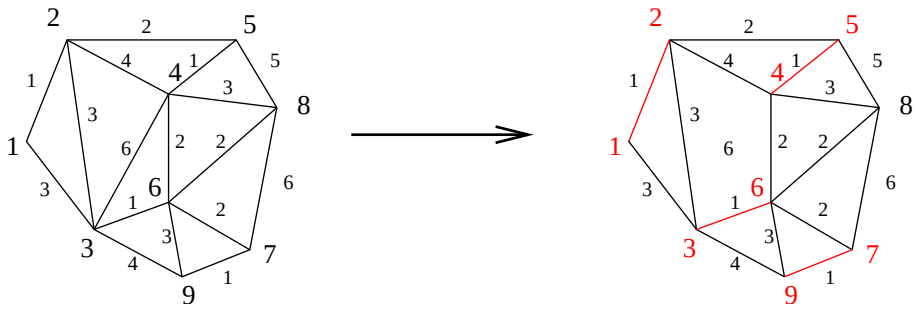


FIG. 2.13: Exemple d'une mise en relation maximale

Cette mise en relation permet de définir les nœuds qui vont être regroupés dans une même entité lors de la génération du graphe de niveau supérieur. Le paragraphe suivant décrit cette opération.

Génération du graphe de niveau supérieur

La difficulté de la définition du graphe supérieur $G_1 = (X_1, A_1)$ réside dans la mise à jour de la valuation des arcs. La figure 2.14(a) présente différents cas de figure qui peuvent être rencontrés :

– les deux nœuds du couple à regrouper sont connectés à un même autre nœud du graphe ;
– les deux nœuds du graphe sont connectés à des nœuds différents du graphe.

La génération de graphes hiérarchiques entraîne fatalement une dégradation de l'information originelle. De ce fait, une recherche sur un graphe de niveau supérieur ne peut avoir la même signification que celle effectuée sur le graphe initial. Le regroupement des nœuds de G_0 les plus proches correspond au regroupement des positions dans l'espace de travail du robot entre lesquels les mouvements sont facilement réalisables. Les nœuds du graphe G_1 ne correspondent donc plus à des positions particulières du système robotique, mais à des zones de l'espace de travail. De ce

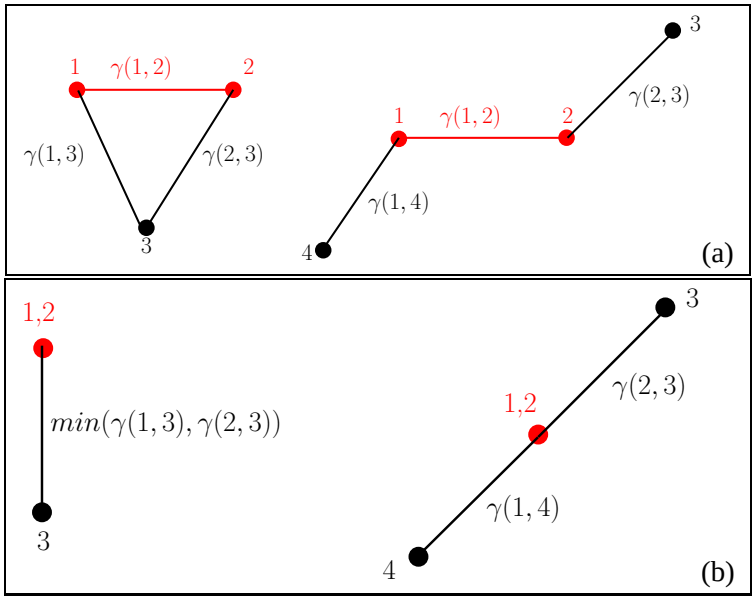


FIG. 2.14: Regroupement de nœuds : (a) différents cas de figure, (b) valuation choisie

fait, les valuations employées dans le graphe G_1 doivent traduire la facilité à se déplacer non pas d'une position à une autre, mais d'une zone à une autre. De cette analyse découle notre stratégie de mise à jour des valuations entre deux nœuds de G_i :

Soit x_i et x_j deux nœuds du graphe réduit G_1. Soit $A' \subset A_0$ l'ensemble des arcs valués de G_0 qui relient un des sommets formant x_i à un des sommets formant x_j. La valuation entre les deux nœuds de G_1 est choisie comme étant la plus faible valeur de l'ensemble A'.

La figure 2.14(b) illustre cette stratégie de mise à jour des valuations. Cette méthode de valuation permet bien de définir la facilité de passer d'une zone à une autre. Notons cependant que les zones ainsi définies ne regroupent pas nécessairement des positions qui sont proches dans l'espace de travail du système robotique. Ces regroupement constituent de proche en proche des trajectoires où le déplacement est aisé.

Ce principe de réduction peut ensuite être appliqué de manière récursive pour générer un graphe de niveau supérieur G_i à partir d'un graphe G_{i-1}. La section suivante présente comment une définition topologique du chemin à parcourir peut être obtenue à partir d'un ensemble $(G_0, \ldots, G_P)$ de graphes hiérarchiques.

2.2.3 Transcription et résolution d'une tâche de navigation

Comme nous l'avons déjà précisé, une tâche de navigation est définie par l'image que doit obtenir la caméra à la fin du déplacement, l'image finale, et l'image qu'elle fournit avant que le déplacement commence, l'image initiale.

La phase de localisation (section 2.2.1) permet d'obtenir pour chacune de ces deux vues celles de la base qui sont visuellement les plus semblables. Le mouvement partiel entre l'image initiale

(ou finale) et chacune de ses images proches est estimé à partir des mises en correspondances détectées entre ces deux vues. De ce mouvement peut ensuite être déduit le coût des arcs permettant de connecter le nœud associé à l'image initiale (ou finale) avec le reste du graphe initial G_0.

Dans le texte qui suit, seule l'image la plus proche de la vue initiale (ou finale) est considérée. Le formalisme employé peut facilement se généraliser à la considération des autres images proches.

L'algorithme de recherche de chemins à partir de graphes hiérarchiques est décrit succinctement dans la table 2. Dans cette algorithme, nous supposons connaître les mises en relation effectuées sur chacun des graphes hiérarchiques. Les fonctions utilisées sont les suivantes (ces fonctions sont illustrées sur la figure 2.15) :

- InclusDans(j, x_i) : retourne le nœud du graphe G_j dans lequel est inclus le nœud x_i de G_0 ;
- PlusCourtChemin(G, x_0, x^*) : extrait du graphe G le plus court chemin, au sens de la fonction de coût, entre les sommets x_0 et x^* ;
- Éclate(Γ_i) : génère un graphe réduit de G_{i-1}, dont les nœuds sont limités à ceux formant les différents sommets de G_i qui sont dans le chemin Γ_i.

[Initialisation :]
$G'_p = G_p$
[Boucle de traitement :]
Pour i **de** p **à** 1 **faire**
  x_{i_0} = InclusDans(i, x_0) ;
  x_{i^*} = InclusDans(i, x^*) ;
  Γ_i = PlusCourtChemin(G'_i, x_{i_0}, x_{i^*}) ;
  G'_{i-1} = Éclate(Γ_i) ;
Fin Pour

Table 2: Algorithme récursif de recherche de chemin

Le chemin final est alors obtenu en réappliquant la recherche du chemin de coût minimum sur le graphe obtenu G'_0.

Notons cependant que la perte d'information induite par la création des graphes hiérarchiques, peut entraîner le choix d'un chemin dans G_i qui ne contient pas le plus court chemin dans G_{i-1}. La hiérarchisation effectuée permet d'assurer une minimisation du coût entre les différents lieux, mais ne permet pas de prendre en compte le coût des déplacements dans un même lieu (un lieu étant un nœud d'un graphe G_i, avec $i \geq 1$ et qui représente donc plusieurs vues de la scène).

Cette particularité ne nous apparaît cependant pas préjudiciable : à chaque niveau du graphe, seuls les nœuds qui sont mutuellement les plus proches sont regroupés. De ce fait, même si le chemin obtenu ne correspond pas au chemin supposé optimal, il décrit tout de même un déplacement facilement réalisable.

Il est cependant possible de minimiser cette déviation en relâchant les contraintes lors de la création du graphe G'_{i-1} à partir de Γ_i. Par exemple, il est possible de rajouter aux nœuds du chemin les sommets de G_i qui sont connectés à au moins deux nœuds de Γ_i.

La section suivante présente les différentes opérations à effectuer pour organiser la base d'images (étape réalisée hors-ligne), et pour répondre à une tâche de navigation (traitement effectué en-ligne).

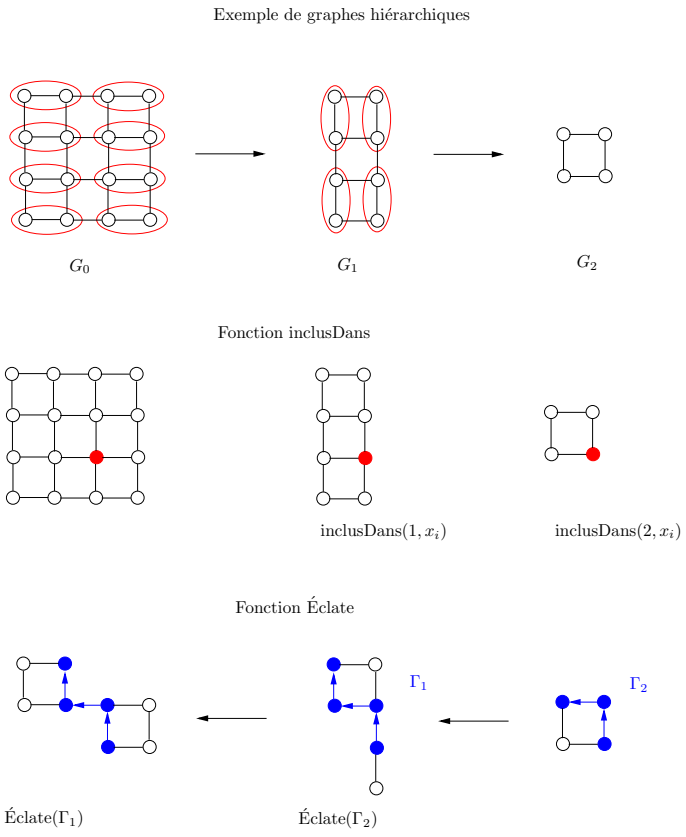


FIG. 2.15: Illustration des fonctions de la recherche de chemin

2.2.4 Schémas temporels récapitulatifs

La figure 2.16 présente les opérations réalisées lors de la génération de la représentation de l'environnement. La sémantique utilisée sur ce schéma est la suivante :

- les rectangles désignent les opérations de traitement ;
- les cercles désignent les résultats des diverses opérations qui sont sauvegardées dans la représentation de l'environnement ;
- les flèches désignent les dépendances entre les opérations.

Finalement, les informations sauvegardées dans la base sont :

- **les images de la base :** elles sont notées ψ_i ;
- **les points d'intérêt :** à chaque image ψ_i est associé un fichier de points ${}^i\mathbf{x}_{p,j}$, où j est un indice permettant de référencer tous les points de la base. Un point est décrit par ses coordonnées pixelliques, ainsi que son score obtenu au détecteur de points. Tous les points sont calculés pour une même taille de support ;
- **les mises en correspondance :** les points communs aux images ψ_i et ψ_j sont regroupés dans l'ensemble $\mathcal{M}_{i,j}$. De manière pratique, un fichier de correspondances contient un ensemble de couples d'indices renvoyant aux deux fichiers de points concernés ;
- **le fichier d'indexation :** il regroupe l'ensemble des descripteurs photométriques calculés pour tous les points de la base. Chaque descripteur contient un lien vers son image d'origine ;

– **les graphes hiérarchiques :** chaque graphe est défini par l'ensemble des arcs valués le constituant. De plus, les tables résultats des phases successives de mise en relation sont sauvegardées afin de pouvoir voyager sans souci dans les différents niveaux de graphe. Au graphe initial G_0 est associé un fichier permettant d'établir les correspondances entre les nœuds de ce graphe et les images de la base.

La figure 2.17 présente les opérations effectuées en-ligne lors de la définition d'une tâche de navigation. La sémantique utilisée est la même que celle du schéma précédent. Nous considérons ici le traitement de l'image initiale. Le traitement de l'image finale est identique (notons que dans une application réelle, l'image à atteindre, puisqu'elle est connue, doit sans doute être une des images de la base).

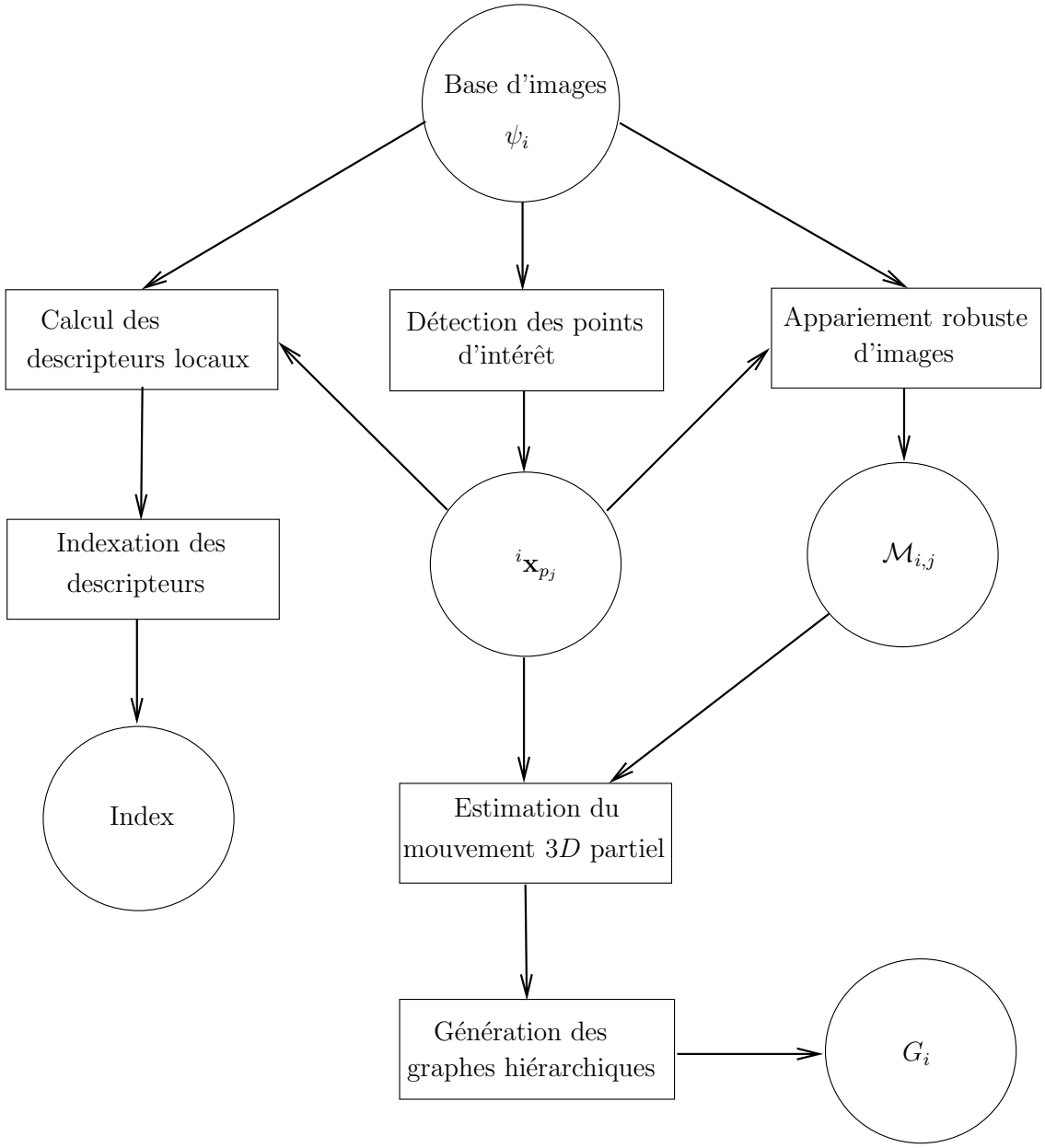


FIG. 2.16: Génération de la représentation interne de l'environnement (hors-ligne)

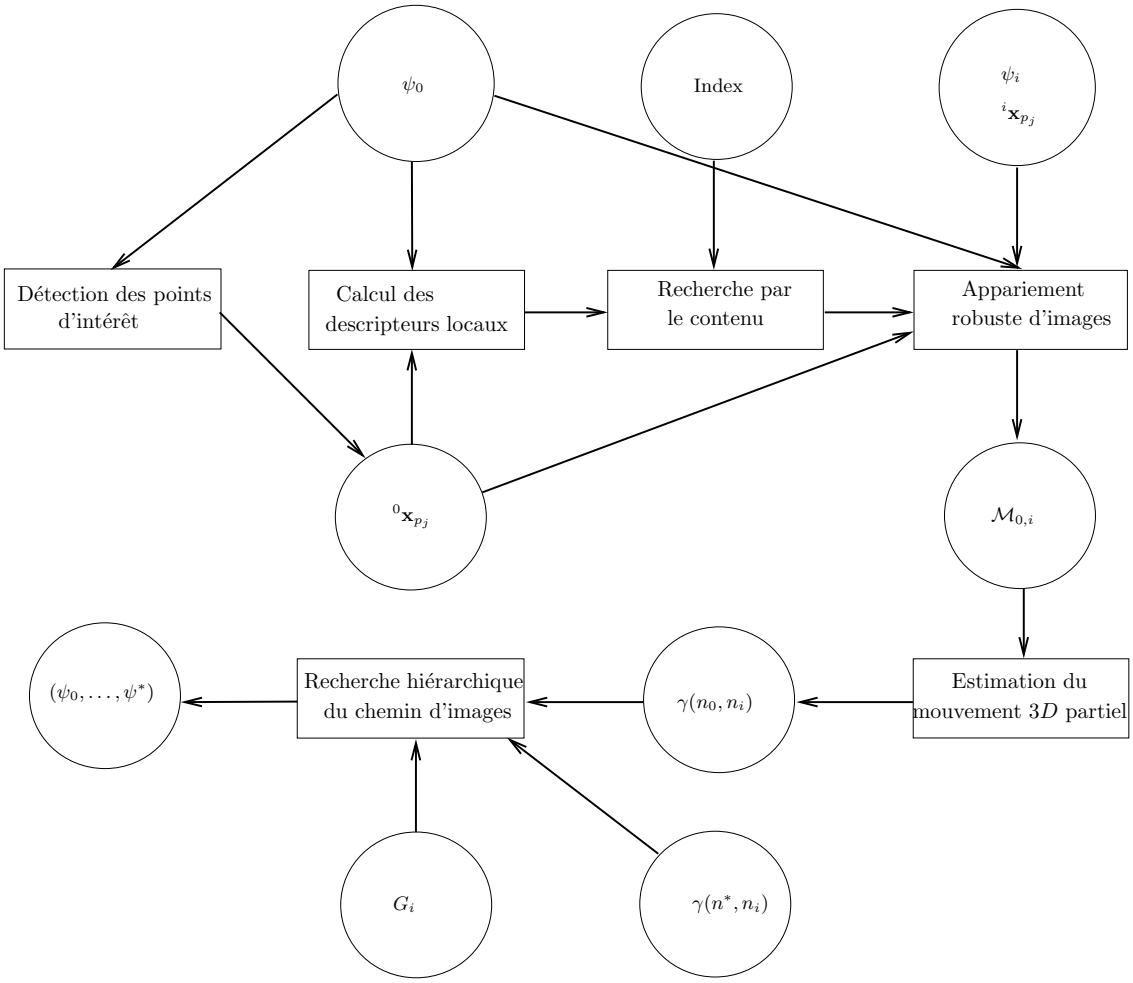


FIG. 2.17: Transcription d'une tâche de navigation en un chemin d'image : $(\psi_0, \ldots, \psi*)$ (traitement en-ligne)

2.2.5 Résultats expérimentaux

2.2.5.1 Recherche de chemin classique

Dans cette première expérience, nous considérons la fonction de coût basée sur le nombre de mises en correspondance, présentée en 2.2.2.1. La recherche est ici directement effectuée sur le graphe G_0. La scène considérée est plane. La valuation utilisée permet d'obtenir un chemin d'images formant une description continue de l'environnement, comme on peut l'observer sur la figure 2.18.

FIG. 2.18: Exemple de chemin d'images. (a) est l'image initiale, et (h) l'image désirée. Les autres images ont été extraites automatiquement de la base.

Cette méthode de recherche de chemin est satisfaisante dans le cas de système robotique se déplaçant parallèlement à un plan (comme par exemple un sous-marin dont la caméra est orientée vers le fond marin). Dans le cas d'un système robotique se déplaçant dans un environnement $3D$, il est préférable d'utiliser la fonction de coût basée sur le mouvement.

2.2.5.2 Valuation basée sur le mouvement

Cette section s'intéresse à la valuation basée sur le mouvement relatif entre les différentes prises de vue formant la mémoire visuelle du système robotique.

La base d'images considérée correspond à une séquence d'images acquise par un système stéréoscopique embarqué dans une automobile. Afin de se concentrer sur la validation de la fonction

de coût, nous nous sommes limités au calcul du mouvement à partir des positions associées aux différentes prises de vue. Ces informations de position nous ont été fournies par Éric Royer, qui dans ses travaux réalise la reconstruction d'une scène à partir d'une séquence d'images [Royer 04]. La figure 2.19 présente la position associée aux différentes images de la séquence. La figure 2.20 illustre cette séquence par quelques vues dont les positions sont indiquées sur le schéma 2.19.

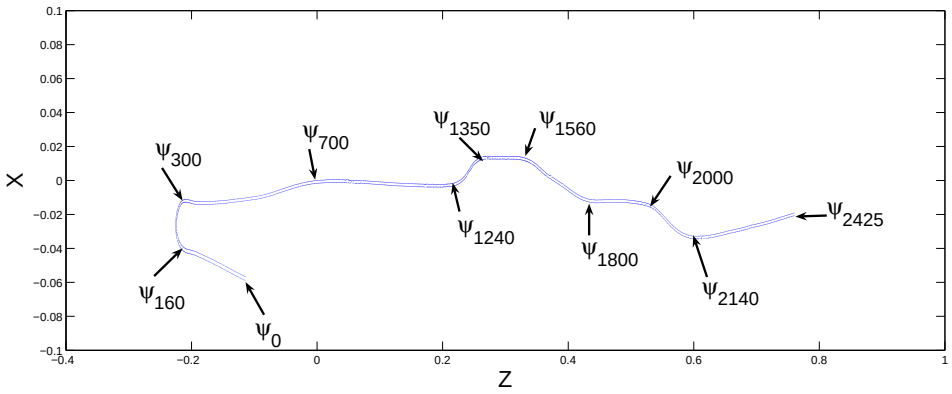


FIG. 2.19: Visualisation de la séquence d'images : positions des différentes prises de vue.

Dans les expériences suivantes, cette séquence d'images va constituer la mémoire visuelle du système robotique. Bien que les images acquises par les caméras de droite et de gauche soient très proches, les considérer ensemble permet de simuler la co-existence de deux trajectoires possibles dans la base. De plus, il serait nécessaire dans une application réelle de sous-échantillonner une telle séquence, le mouvement inter-image étant très faible. Dans les expériences présentées, nous avons choisis de toutes les conserver, afin d'observer le comportement de notre approche face à une base de taille conséquente.

La figure 2.22 présente les valuations obtenues sur une portion de la séquence (des nœuds x_0 à x_{430}). Les différentes courbes représentent les coûts des arcs d'un sommet x_i (correspondant à une image de la caméra gauche) vers les nœuds x_{i+2} à x_{i+7}. Les sommets d'indice pair sont associés à des images prises par la caméra de gauche, comme l'illustre le schéma 2.21 (le mouvement entre les nœuds x_i et x_{i+1} étant quasiment une translation pure sur l'axe $\vec{x}$, la valuation associée est très forte ; pour garder une échelle lisible, ces valuations ne figurent pas sur le schéma 2.22). Pour la figure 2.22(a) (respectivement 2.22(b)), le terme κ utilisé dans la fonction de coût (équation (2.1) page 53) vaut 0.01 (resp. 10).

Notons dès à présent que dans les deux cas, la forme générale de la fonction reste identique. Une grande valeur de κ augmente simplement l'amplitude de variation de cette fonction de coût. Les intervalles $[0, 150]$ et $[350, 430]$ correspondent à un déplacement le long de l'axe optique de la caméra. Le coût des arcs est, comme espéré, plus faible vers les vues acquises par la même caméra. L'intervalle $[150, 350]$ correspond au virage à 180 degrés réalisé par le véhicule. On observe alors qu'un coût plus faible est obtenu avec les images acquises par la caméra de droite. Il est en effet dans ce cas plus aisé de se déplacer vers ces nœuds : le mouvement de x_i vers x_{i+5} ou x_{i+7} présente alors une rotation moins importante, et accompagné d'une plus grande translation le long de l'axe optique.

FIG. 2.20: Visualisation de quelques images de la séquence considérée.

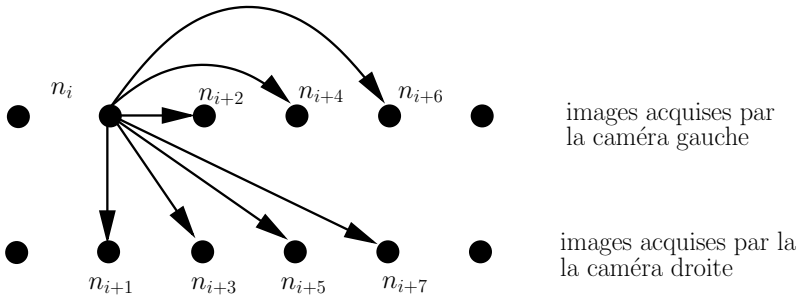
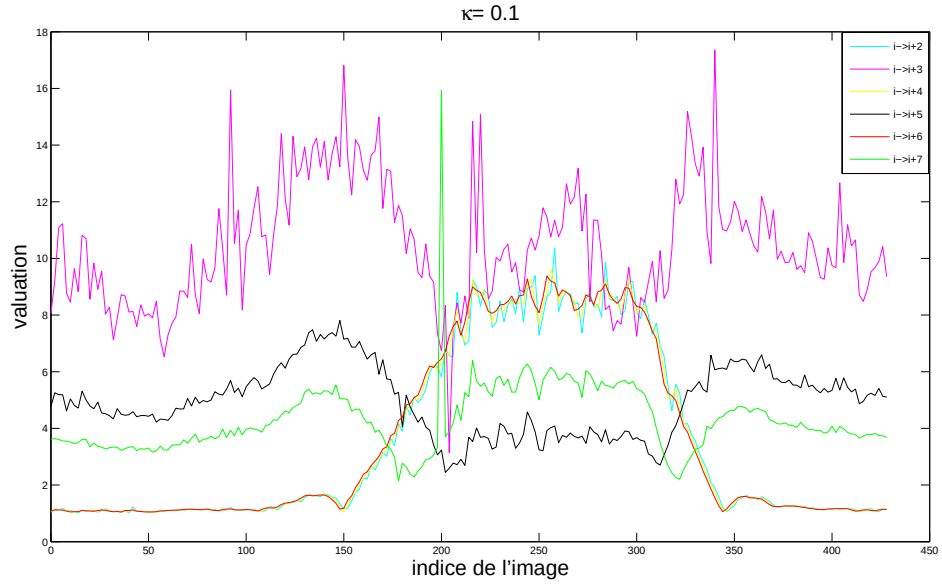
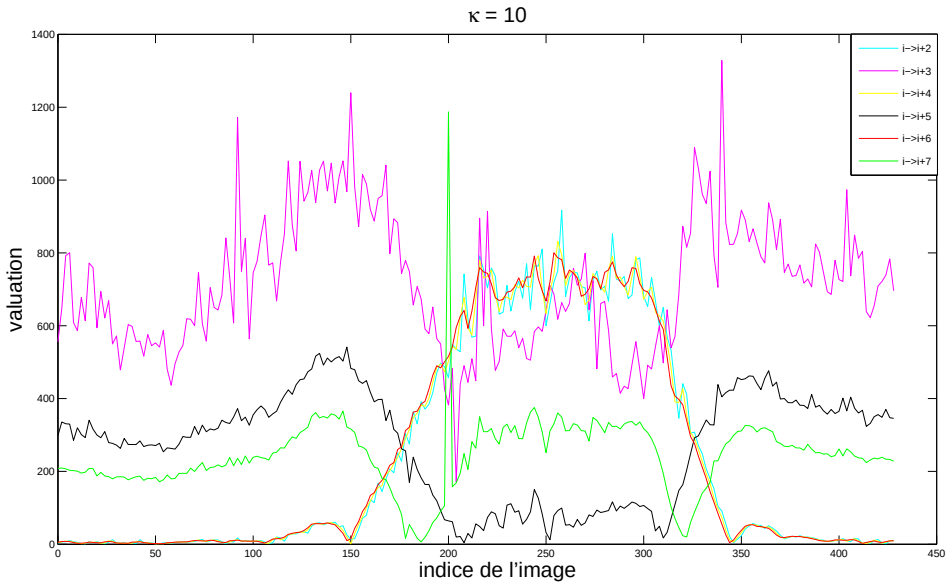


FIG. 2.21: Visualisation des connections « vers l'avant » d'un nœud x_i du graphe.



(a)



(b)

FIG. 2.22: Coût des déplacements inter-image, pour (a) $\kappa = 0.1$, et (b) $\kappa = 10$. L'abscisse correspond à l'indice de l'image, prise par la caméra de gauche. La valeur de la valuation est tracée en ordonnée. Les valuations sont calculées vers différentes vues, acquises soient par la même caméra (courbes bleu, rouge et jaune), soient par la caméra de droite (en vert, noir et rose).

Le pic observé sur la courbe verte, aux alentours de l'indice 200, est dû à une mauvaise estimation de la position d'une des caméras. La forte discontinuité observé montre qu'il pourrait être envisagé de détecter les erreurs d'estimation de mouvement relatif, en étudiant la continuité de ces différentes courbes.

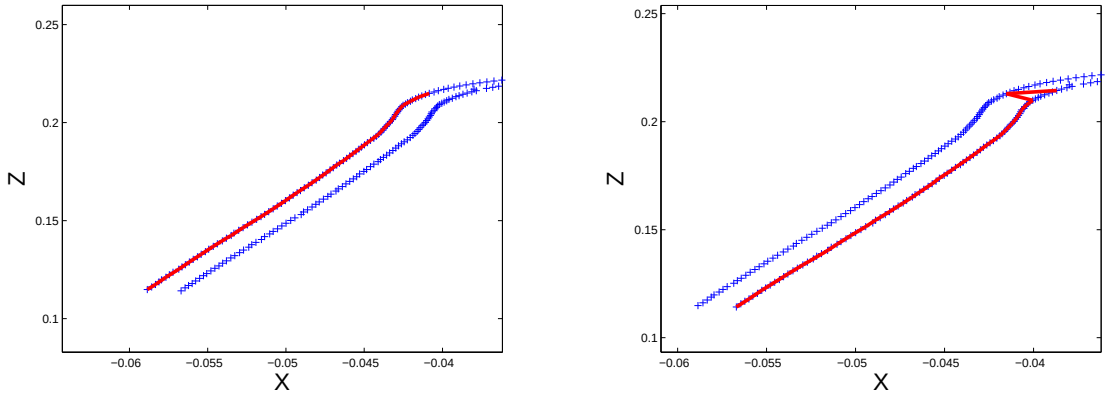
Dans les expériences de recherche de chemin qui suivent, un nœud x_i du graphe (associé à une image de la base) est considéré comme connecté à treize autre nœuds $(x_{i-6}, \ldots, x_{i+7})$. Les vues d'indice pairs (respectivement impairs) sont celles acquises par la caméra de gauche (resp. de droite). Le graphe total est constitué de 2426 nœuds.

Les figures 2.23, 2.24 et 2.25 présentent des résultats de recherche de chemin sur différents zones de la séquence totale. Sur chacune de ces zones, un chemin est recherché entre deux images acquises par la caméra de gauche et de droite.

Sur la figure 2.23(b), une image acquise par la caméra de gauche a été sélectionnée dans le chemin. Ce changement permet de minimiser le coût global du chemin, en minimisant la rotation que doit réaliser le système robotique, comme nous l'avions déjà noté sur les schémas de la fonction de coût. Naturellement, si la stratégie employée pour déplacer le robot consiste à atteindre successivement les différentes positions intermédiaires, ce comportement peut être préjudiciable. Par contre, il ne l'est pas si ces images ne sont pas associées à des poses cibles, mais simplement comme des descriptions locales de l'environnement à traverser, comme c'est le cas ici. Notons enfin que ce comportement peut de plus être attribué à la forte rotation qui est réalisée par le véhicule. La figure 2.24 présente un autre chemin obtenu sur cette séquence. Bien que cette zone présente des mouvements de rotation, les trajectoires obtenues ne contiennent que des images acquises par la même caméra.

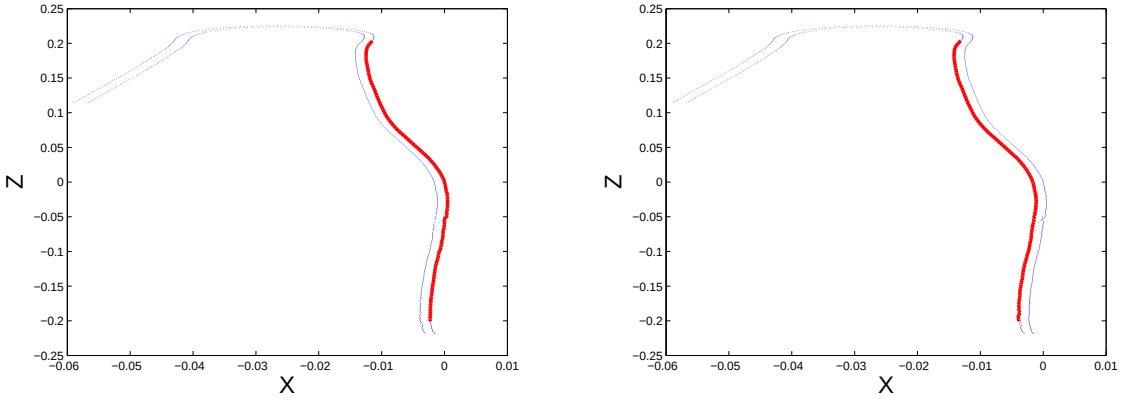
La figure 2.25 s'intéresse au virage complet du début de la séquence. Le changement de caméra (en passant du nœud x_i à x_{i+7}) permet encore une fois de minimiser l'amplitude de la rotation entre les images, et donc de diminuer par la même occasion le coût global du chemin.

Notons enfin que la valuation utilisée permet non seulement de minimiser le coût global du chemin, mais aussi de minimiser le nombre de sommets le constituant. Par exemple, dans le cas d'une séquence de translation pure, il coûtera moins cher d'aller directement du nœud x_i au sommet x_{i+2} que de passer par le nœud intermédiaire x_{i+1}. La table 2.26 présente pour chacun des chemins considérés ci-dessus le nombre de nœuds sélectionné, ainsi que le coût global. Dans chacun des chemin, le nombre de noeuds est moins important que le nombre de vues entre les positions initiale et désirée. On observe de plus que la réalisation du virage est beaucoup plus coûteuse que les autres déplacements (même si elle est composée par exemple de moins de nœuds que les chemins 2(a) et 2(b)).



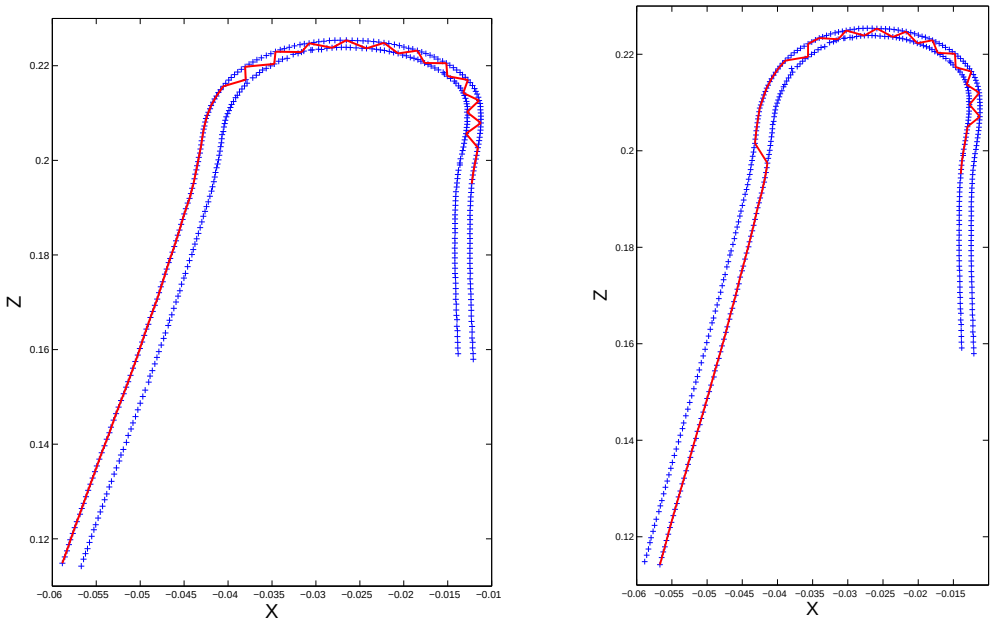
(a) (b)

FIG. 2.23: Chemin 1, sur le début de la séquence (entre les sommets 0 et 176 pour la figure (a), 1 et 177 pour (b)). La ligne rouge correspond au chemin d'images sélectionné.



(a) (b)

FIG. 2.24: Chemin 2, au milieu de la séquence (des sommets 330 à 1150 et 331 à 1151).



(a) (b)

FIG. 2.25: Chemin 3, cas d'un grand virage (des nœuds 0 à 350 et 1 à 351).

Chemin	Nombre d'images dans le graphe	Nombre d'images sélectionnées	Coût global
1(a) (0 à 176)	83	31	40
1(b) (1 à 177)	83	31	40
2(a) (330 à 1150)	400	138	160
2(b) (331 à 1151)	400	139	159
3(a) (0 à 350)	175	56	215
3(b) (1 à 351)	175	56	215

FIG. 2.26: Caractéristiques des chemins de nœuds obtenus. Le nombre d'images dans le graphe correspond au nombre de prises de vue se situant entre les deux vues d'intérêt dans la séquence associée.

2.2.5.3 Recherche de chemins avec les graphes hiérarchiques

Les premiers résultats présentés viennent illustrer la méthode de génération des graphes hiérarchiques ainsi que la technique de recherche de chemins à partir de ces graphes. La figure 2.27 décrit la base d'images qui est considérée. Dans le graphe associé, les nœuds de la ligne droite sont connectés « vers l'avant » vers les quatres nœuds suivants. Dans la zone contenant un virage, les noeuds sont connectés vers les deux noeuds suivants.

Sur la figure 2.28 sont présentés les différents graphes hiérarchiques qui peuvent être générés à partir du graphe initial G_0. Les nœuds les plus proches sont regroupés dans une même entité dans le graphe de niveau supérieur. On observe que les nœuds correspondant à un déplacement le long de l'axe optique sont mis en relation très rapidement. Par contre, les zones où le mouvement entre les images comprend une rotation sont plus lentement regroupées. En effet, le déplacement est alors considéré comme plus difficile. Cette lenteur de regroupement est aussi dû au fait que la mise en relation ne peut s'effectuer que si les deux nœuds se considèrent mutuellement comme les plus proches. Ainsi, si l'on nomme x_j le nœud le plus proche de x_i, le regroupement ne peut s'effectuer que si x_j considère x_i comme étant son sommet le plus proche. Si ce n'est pas le cas, le regroupement ne peut se réaliser à ce niveau hiérarchique. Une définition plus souple de la méthode de mise en relation permettrait de rendre plus rapide ces regroupements.

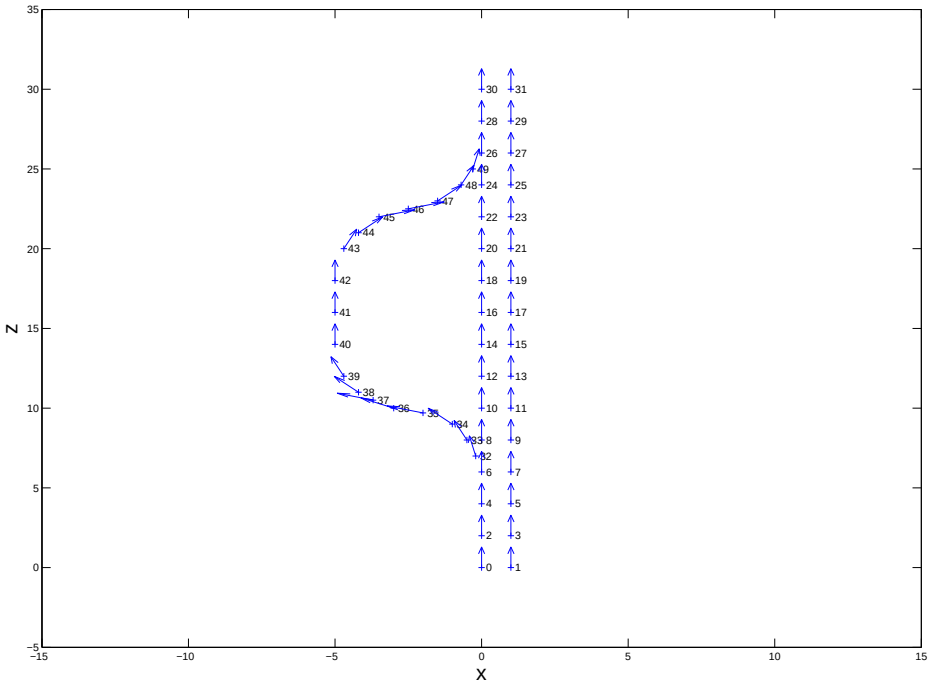


FIG. 2.27: Exemple d'une base d'images composée de plusieurs chemins. Les flèches indiquent les orientations des caméras.

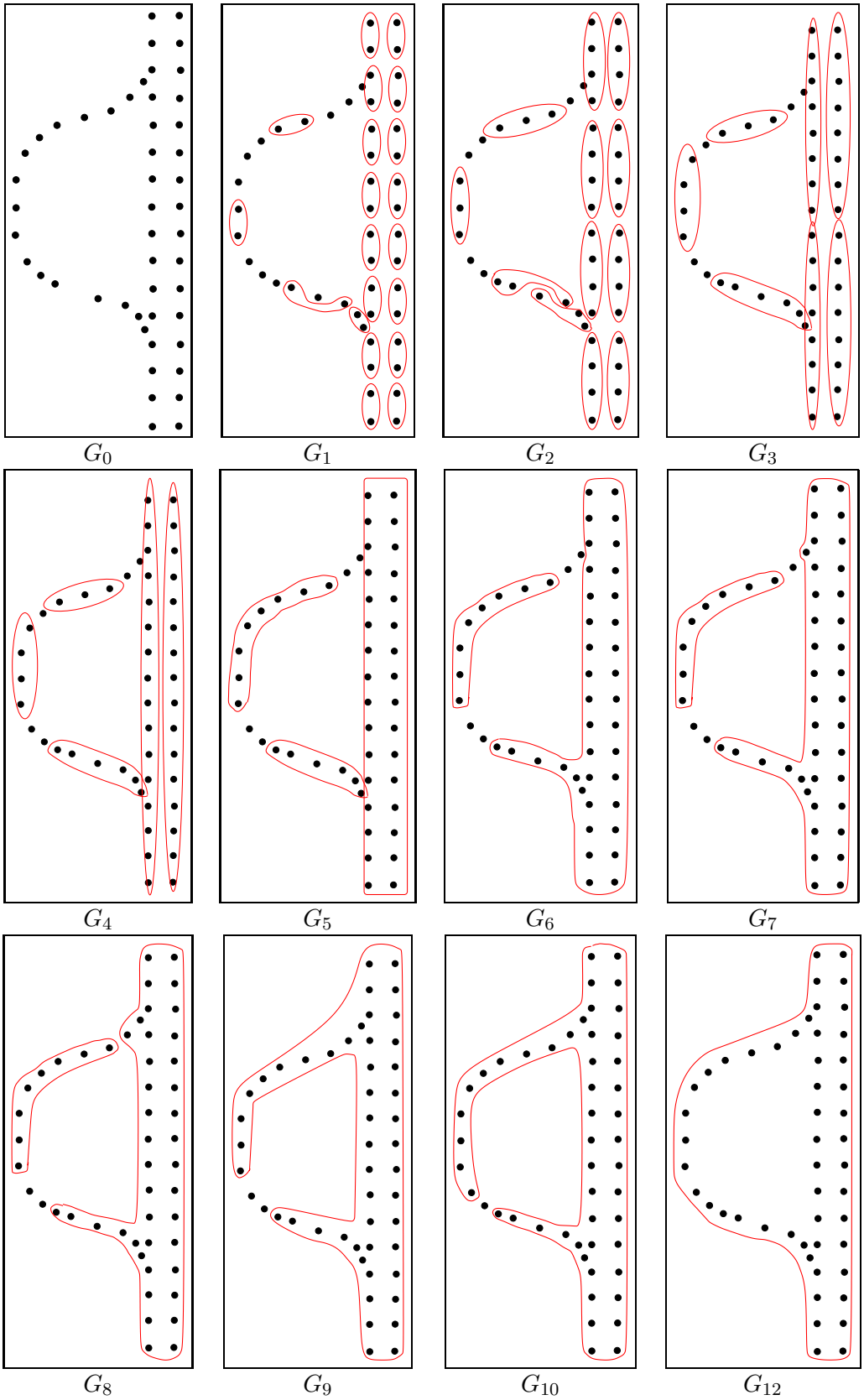


FIG. 2.28: Génération des graphes hiérarchiques. Les ensembles rouges correspondent aux nœuds de G_0 qui sont regroupés dans une même entité dans le graphe G_i.

Un exemple de recherche de chemin sur ces graphes est présenté sur la figure 2.29. La recherche est ici considérée à partir du graphe G_4, entre les sommets 0 et 31. Dans G_4, la recherche s'effectue entre les deux nœuds contenant ces sommets de G_0. Les noeuds constituant le chemin sont dessinés en vert. À partir des informations de mise en relation ayant permis de créer G_4, il est possible de définir un graphe réduit G'_3, limité aux sommets constituant Γ_4. La recherche de chemin s'effectue alors sur ce graphe. De proche en proche, on obtient un graphe réduit G_0 de taille moindre par rapport à G_0. Le dernier schéma de la figure présente le chemin calculé sur ce graphe.

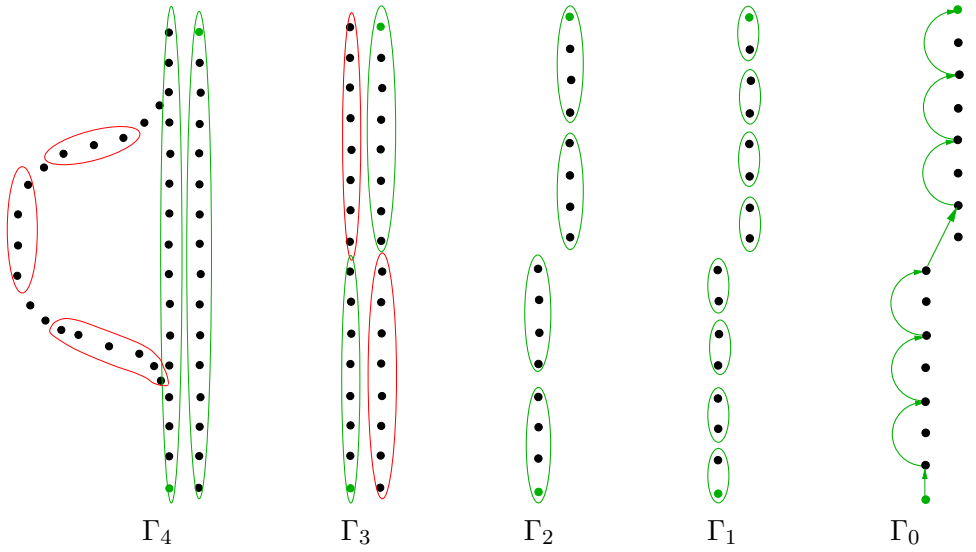


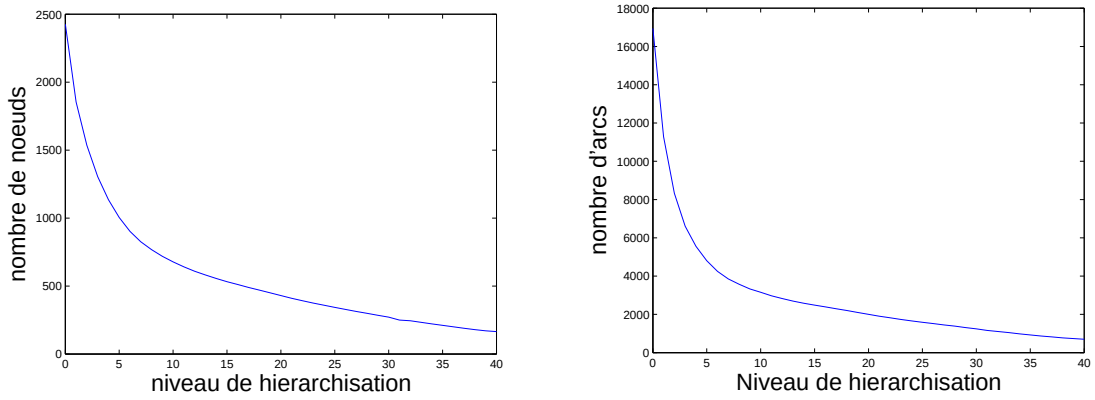
FIG. 2.29: Recherche de chemin sur les graphes hiérarchiques, entre les nœuds 0 et 31. Les nœuds verts correspondent au chemin de coût minimum obtenu à chaque niveau hiérarchique.

Nous pouvons relever une autre limitation de notre méthode actuelle de valuation. En effet, nous générons un graphe non orienté : le coût pour passer d'un nœud x_i à x_j est le même que celui pour passer de x_j à x_i. Il n'y a donc pas de notion d'ordre de passage entre les nœuds, alors que dans le cas d'un robot mobile, cette contrainte est importante. En effet, si l'on observe le graphe de niveau 9 de la figure 2.28, nous pouvons en déduire que la recherche d'un chemin entre les nœuds 0 et 44 de G_0 par exemple fournirait une trajectoire qui passerait par le nœud 49, alors que ce chemin est censé être parcouru en passant par le nœud 32. Il serait donc nécessaire à l'avenir de prendre en compte le sens de parcours des images de la séquence lors de la création du graphe et du calcul du coût des arcs, afin d'éviter ces cas de figure.

Les expériences suivantes ont été réalisées sur la séquence utilisée pour valider la fonction de coût. Quarante niveaux de hiérarchisation ont été générés. Comme la figure 2.30 le montre, la génération de ces différents graphes permet de diminuer considérablement la taille du graphe.

Le graphe de niveau 40 est finalement constitué de 165 nœuds, soit une réduction d'environ 93% de la taille du graphe. Comme nous l'avons déjà mentionné, les zones où le déplacement

est aisé sont beaucoup plus facilement mises en relation et regroupées. Le schéma 2.31 montre en effet que cette propriété est retrouvée lors du traitement de la séquence d'images. Les traits présentés sur ce schéma représentent les zones couvertes par 13 de ces noeuds. Ceux-ci représentent environ 91% de la séquence d'images.



(a) (b)

FIG. 2.30: Évolution du nombre de noeuds (a) et du nombre d'arcs (b) dans les différents graphes hiérarchiques successifs.

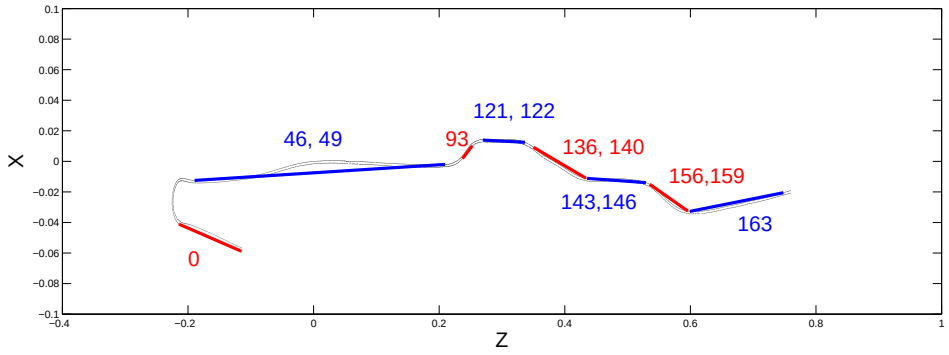
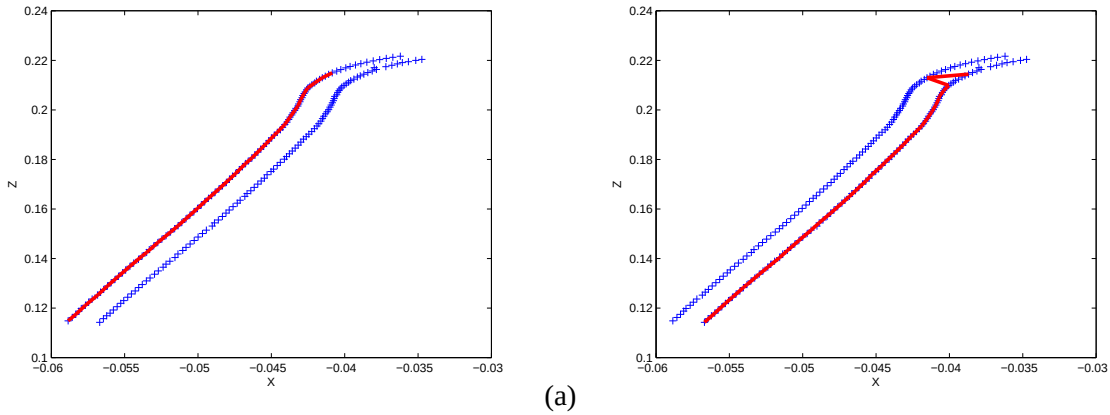


FIG. 2.31: Principaux nœuds de G_{40}. Les lignes correspondent aux zones contenues dans un nœud. Celles associées à deux indices de nœuds sont des zones où un sommet regroupe la séquence acquise par la caméra de gauche, et l'autre celle acquise par la caméra de droite (les couleurs ont pour seul but de faciliter la distinction entre les différentes zones).

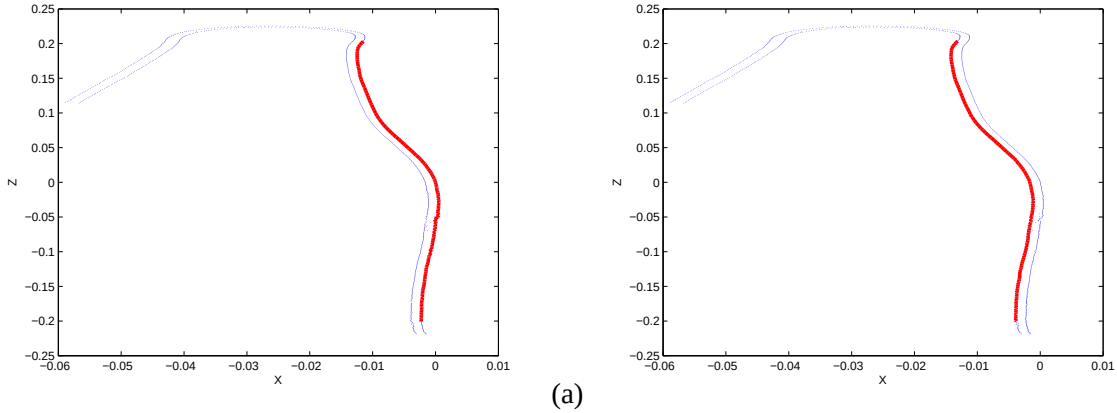
Les figures 2.32, 2.33 et 2.34 présentent des recherches de chemin réalisées à partir de ces graphes hiérarchiques. Les trajectoires obtenues sont analogues à celles calculées directement à partir de G_0. La perte d'information lors de la génération des graphes hiérarchiques ne permet cependant pas d'obtenir exactement les mêmes trajectoires (la figure 2.35 décrit plus en détail les différents chemins obtenus). En effet, le graphe réduit G'_0 sur lequel s'effectue la dernière recherche ne contient pas tous les nœuds existant entre le sommet initial et désirée, et certains nœuds choisis lors d'une recherche directement sur G_0 peuvent ne pas être présents. Cependant, si le coût global augmente, le coût des déplacements entre les nœuds successifs de ce chemin reste très faible.

Enfin, la perte de qualité de la recherche est compensée par la simplification du graphe. Si l'on considère par exemple le chemin recherché sur la figure 2.32, la taille du graphe à considérer passe dès la recherche dans G_{40} à moins de 200 nœuds. La recherche sur les graphes de niveau inférieur s'effectue donc sur des graphes plus petits, restreints à la zone du graphe succeptible de contenir un chemin entre les deux d'intérêt de G_0.



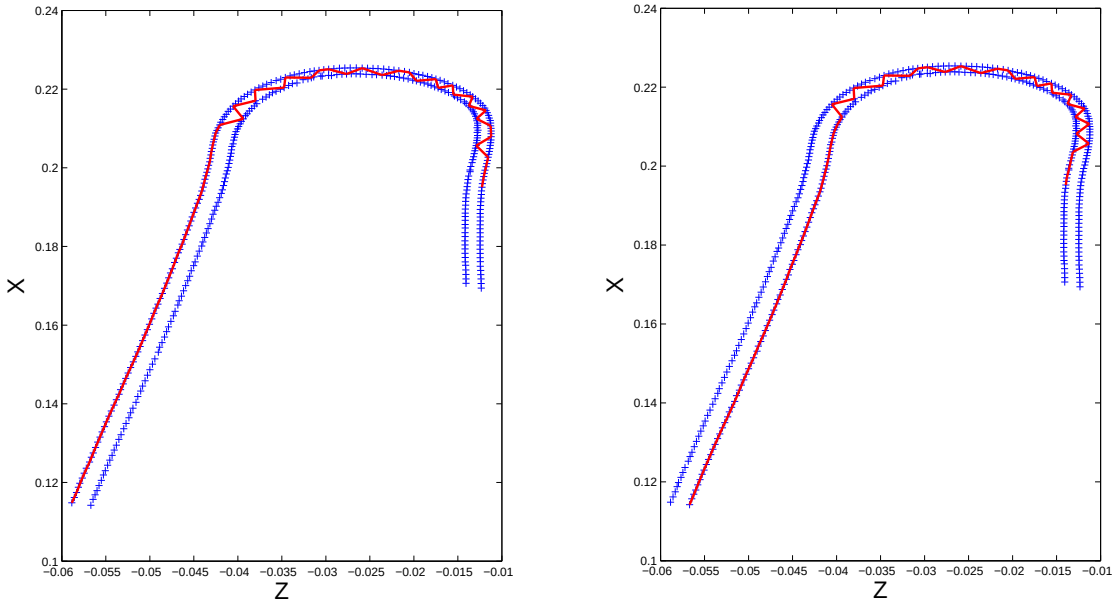
(a) (b)

FIG. 2.32: Chemin obtenu dans G_0, après recherche dans les graphes hiérarchiques (des nœuds 0 à 176 et 1 à 177.



(a) (b)

FIG. 2.33: Chemins des noeuds 330 à 1150 et de 331 à 1151



(a) (b)

FIG. 2.34: Chemins des sommets 0 à 350 et de 1 à 351

Chemin	Nombre d'images	Coût global
1(a) (0 à 176)	34 (31)	44 (40)
1(b) (1 à 177)	35 (31)	47 (40)
2(a) (330 à 1150)	158 (138)	183 (160)
2(b) (331 à 1151)	160 (139)	182 (159)
3(a) (0 à 350)	63 (56)	246 (215)
3(b) (1 à 351)	64 (56)	241 (215)

FIG. 2.35: Caractéristiques des chemins de nœuds obtenus. Entre parenthèses sont rappelés les résultats obtenus lors des recherches correspondantes directement sur G_0.

2.3 Conclusion

Pour qu'un système robotique puisse « librement » naviguer dans un environnement, il est nécessaire de lui fournir la capacité de se représenter cet espace, non seulement pour se localiser, mais aussi pour avoir la capacité à se définir des chemins dans celui-ci.

Il existe différentes familles de représentation d'un environnement. Le choix d'un formalisme est guidé par les connaissances que l'on se donne sur l'environnement, et par les contraintes que l'on pose sur les comportements du système robotique durant la phase de navigation pure.

Le formalisme proposé découle de trois propriétés que nous voulions donner au système global. La première consiste à s'affranchir de l'utilisation d'un modèle $3D$ de la scène, tant pour représenter l'environnement, que pour guider les mouvements du robot. Un tel modèle n'est pas toujours disponible, et la construction (ainsi que sa maintenance au cours du temps) de celui-ci reste encore un problème ouvert.

La deuxième propriété est de laisser à la phase de navigation un large pouvoir décisionnel, afin que le système robotique puisse au mieux s'adapter, lors de ces mouvements, aux différents changements et/ou perturbations que peut subir l'environnement.

Enfin, la dernière caractéristique recherchée est celle d'un système centré autour de la vision afin de profiter de la richesse de l'information acquise par ce type de capteur.

C'est pourquoi nous avons établi une représentation d'un environnement de navigation basée sur l'apparence. L'espace dans lequel doit évoluer le système robotique est décrit par une base d'images acquise lors d'une phase d'apprentissage.

Cette mémoire visuelle est organisée sous la forme d'un graphe topologique. Les contraintes de mouvement du système robotique sont prises en compte dès la construction de se graphe. Nous nous assurons ainsi de fournir à la phase de navigation les informations visuelles qui sont les plus susceptibles de caractériser la zone de l'espace que le robot est censé traverser.

Afin de compenser les problèmes liés à la taille de la base d'information, nous avons proposé une structure hiérarchique de ce graphe topologique. Cette structure permet, à chaque niveau hiérarchique et de manière descendante, de limiter la quantité d'information à traiter pour trouver le chemin d'images.

La définition d'un chemin de navigation passe par la localisation du système robotique. Dans une approche basée vision, ce problème est très proche de la reconnaissance d'images, et est de ce fait généralement traitée en utilisant le savoir-faire du domaine de la recherche d'images. La technique que nous avons utilisée est basée sur une description locale des images. Son application à la robotique permet des localisations très satisfaisantes en environnement extérieur, tout en autorisant de considérer des bases d'images de taille conséquente.

De nombreuses perspectives peuvent être données à ces travaux. En ce qui concerne la phase de localisation, nous n'avons pas cherché dans notre approche à optimiser le temps de recherche. Une solution, entre autres, serait encore une fois de s'appuyer sur le savoir-faire de la communauté de reconnaissance d'images. Citons par exemple les travaux de Berrani [Berrani 03], dans lequel les descripteurs sont regroupés par rapport à leur proximité dans l'espace, grâce à des techniques de clustering. Il a d'ailleurs démontré qu'autoriser une faible approximation lors de la recherche des plus proches voisins des descripteurs image permet de diminuer fortement le volume de ces différents clusters. Cette recherche est alors beaucoup plus rapide, tout en préservant une grande qualité de reconnaissance.

L'adjonction d'information topologique dans la représentation de l'environnement permettrait d'accélérer et de simplifier grandement la recherche de chemin. On peut ainsi penser à une organisation de la base en fonction d'entités de plus haut niveau, comme des routes ou des quartiers. La recherche de chemin n'aurait plus alors qu'à se faire à l'intérieur de ces zones.

La base d'images est censée fournir au système robotique une description la plus proche possible de l'environnement dans lequel celui-ci va se déplacer. Or l'aspect visuel d'un environnement varie très fortement ne serait-ce qu'en fonction du moment de la journée pendant lequel on l'observe. Il semble donc nécessaire d'avoir à disposition dans la base d'images plusieurs vues pour une même position dans l'espace, chacune étant fonction de conditions extérieures différentes.

Il est donc nécessaire de définir et mettre en place une base de données multi-modale permettant de sélectionner dans la mémoire visuelle les informations qui sont les plus proches des conditions extérieures dans lesquelles le système robotique va devoir naviguer.

Nous n'avons pas du tout considéré la phase d'apprentissage, qui n'est pourtant pas triviale. En effet, si l'on désire travailler en environnement urbain, il est illusoire de penser que la phase d'apprentissage pourra s'effectuer sans la présence d'objets dynamiques (piétons, autres véhicules, ...). Ces objets ne constituent pas des caractéristiques propres de la scène, qui est elle statique. La phase d'apprentissage devra donc nécessairement prendre en compte ces contraintes. On peut alors penser mettre en pratique les recherches effectuées en détection de mouvement [Odobez 98]. En effet, puisque l'environnement à caractériser est statique, le déplacement dominant observé dans l'image est donc simplement dû au mouvement de la caméra. Tous les objets dynamiques de la scène ayant leur propre mouvement, il serait alors possible de les détecter comme des zones des images ne respectant pas le mouvement dominant. La description de ces images pour leur insertion dans la base d'information pourrait alors s'affranchir de la considération de ces zones bruitées.

De plus, l'analyse *a posteriori* d'une tâche de navigation semble importante afin d'assurer la pertinence des informations stockées dans la base. Par exemple, si un ensemble d'amers qui étaient supposés apparaître dans le champ de vision de la caméra n'ont pu être localisés, ce post-traitement permettrait de prendre en considération cette information et de mettre la base à jour en conséquence. Par la même occasion, il serait possible d'obtenir, grâce aux images acquises durant la navigation, de nouveaux amers visuels décrivant des zones nouvelles de la scène, ou bien des zones qui n'ont pu être décrite de manière satisfaisante lors de l'apprentissage (comme des zones qui étaient occultées par les autres véhicules ou encore des piétons).

Le cœur de l'organisation de la mémoire visuelle réside dans la mise en correspondance des images de la base. Or il s'avère que l'algorithme que nous utilisons ne permet pas d'apparier de manière satisfaisante des images acquises suite à un déplacement trop conséquent le long de l'axe optique. La conséquence directe est qu'il est alors nécessaire de conserver dans la base une trop grande quantité de prises de vue.

Une perspective intéressante réside donc dans la prise en considération de points d'intérêt localisés et caractérisés dans un cadre multi-échelle [Mikolajczyk 01, Espiau 02, Lowe 04]. L'utilisation d'algorithmes d'appariement d'images basés sur de tels points permettrait alors d'autoriser de plus amples mouvements entre chaque prise de vue, et par la même occasion de limiter le nombre d'images à stocker dans la base.

Chapitre 3

Mise à jour des amers visibles durant la navigation

Une des caractéristiques des tâches de navigation est l'ampleur du déplacement que doit réaliser le système robotique. La zone où se situe le robot avant le début de ses mouvements peut donc être très éloignée de la zone qu'il doit atteindre. Du point de vue du capteur de vision, l'image initiale (avant le mouvement) peut ainsi être en terme de contenu totalement différente de l'image désirée (associée à la position que doit atteindre le système robotique).

Comme cela a été présenté dans le chapitre précédent, l'utilisation d'une représentation interne de l'environnement devient alors impérative. L'intérêt de cette « mémoire visuelle » est double :

– elle permet de localiser le système robotique, grâce à l'image fournie par sa caméra embarquée ;
– elle permet de définir un « plan de route » afin d'atteindre la position désirée.

D'un point de vue commande, le plan de route fournit les informations permettant de contrôler en ligne les mouvements du robot (quelle que soit la stratégie de navigation employée). Dans une approche basée vision, le plan de route doit pouvoir se traduire dans l'espace associé à ce capteur. Cette propriété permettra, durant la navigation, de définir les mouvements du système robotique, en fonction des informations visuelles extraites de l'image fournie par la caméra et de celles associées au plan de route.

De manière générale, les systèmes robotiques contrôlés par la vision extraient et utilisent un ensemble d'amers des images. Pour assurer le succès de la tâche de navigation, la caméra doit toujours avoir dans son champ de vision un ensemble suffisant de ces primitives. En effet, puisque les mouvements du robot sont déduits de la position dans l'image de ces amers, si aucun des amers connus par le système n'est visible, le robot ne peut plus définir ses mouvements.

Cependant, comme nous l'avons spécifié ci-dessus, l'ampleur du chemin que doit parcourir le robot est telle que les amers qui ont pu être détectés alors que le robot était à sa position initiale ont très peu de chance (si ce n'est aucune) de rester visibles durant tout le déplacement.

Ainsi, le système robotique doit connaître des amers qui sont susceptibles de devenir visibles durant la navigation, afin de pouvoir compenser la disparition de certaines primitives. Plus exactement, il doit avoir la capacité de localiser ces amers lorsqu'ils rentrent dans le champ de vision de sa caméra. Ce problème de mise à jour des primitives visibles est le sujet de ce chapitre.

La première partie présente comment les techniques de navigation basée vision proposées dans la littérature abordent cette mise à jour des primitives visibles durant la navigation. La deuxième partie présente la méthode que nous proposons, dans le cas d'environnements tri-dimensionnels et plans Des résultats viennent ensuite confirmer la validité de cette approche.

3.1 Gestion de l'évolution des amers visibles : état de l'art

Il est clair que, dans le cadre de la navigation autonome, la gestion de l'évolution des amers visibles (tout comme d'ailleurs la représentation choisie de environnement) dépend du schéma de contrôle des mouvements qui est appliqué. Il nous semble cependant important de bien distinguer ces deux opérations. En effet, si le schéma de contrôle du robot est fortement dépendant du *type* d'amers utilisé dans l'application (image complète, ligne, plan, ...), il est *indépendant* de la méthode employée pour localiser ces amers.

De plus, le module de localisation des amers visibles dans la prise de vue courante n'a pas pour seule fonction de fournir des données pour contrôler les mouvements du robot. Dans de nombreux travaux que nous avons cités dans l'application précédente, la localisation du système robotique (avec ou sans tâche de déplacement autonome ensuite) s'effectue à partir des amers extraits de l'image.

Pour bien cadrer l'état de l'art présenté ci-dessous, nous nous plaçons dans le contexte d'un système robotique en mouvement. L'image fournie vient d'être traitée, c'est à dire qu'un ensemble d'amers visuels est connu dans cette image. Une nouvelle image est alors acquise. L'objet de cet état de l'art est de cerner comment les systèmes actuels localisent des amers qui viennent d'entrer dans le champ de vision de la caméra.

3.1.1 Méthodes n'utilisant pas d'amers locaux

De nombreuses méthodes de navigation considèrent l'image acquise comme un amer en soi. Certaines méthodes considèrent directement l'image en niveau de gris [Matsumoto 00b]. D'autres approches réduisent la taille de l'information à traiter en extrayant les principales composantes du gradient de l'image [Košecká 03, Kröse 99], ou encore en définissant des histogrammes [Ulrich 00, Wolf 02, Yuen 02]. La mise en relation de l'amer courant avec les vues de la base s'effectue généralement par une mesure de similarité.

De manière générale, pour que ces amers globaux soient performant le robot est restreint à suivre une trajectoire pré-définie, ou bien à rester proche des positions où les images de la base ont été acquises. Ces techniques n'utilisent en effet que l'apparence des images, sans prendre en compte les informations géométriques induites par celles-ci.

Dans certaines approches, la base n'a pas besoin de contenir d'amers pour que le robot puisse se localiser. C'est le cas par exemple d'un système robotique qui doit avancer dans un couloir [Vassallo 00]. Le système robotique sait alors exactement quelle information chercher dans la nouvelle image (les lignes délimitant le sol du couloir). Cependant le champ d'application de ce type d'approche est restreint à des environnements particuliers, et impose une forte connaissance sur la structure de la scène.

3.1.2 Mise à jour basée sur la connaissance du modèle de la scène

Lorsque l'environnement est décrit par un modèle tri-dimensionnel, la mise à jour des primitives visibles utilise généralement la connaissance de la position du système robotique lors du traitement de l'image précédente. C'est par exemple le cas dans [Cobzas 03]. Dans ce papier, une caméra pan-tilt (pour générer des images panoramiques) est couplée avec un laser, et les amers considérées sont des lignes verticales. La localisation des amers s'effectue en deux étapes. Toutes les lignes verticales de l'images sont tout d'abord extraites avec des techniques standards. La mise en relation des amers de l'image avec ceux de la base est réalisée par une méthode de relaxation basée sur la mesure de Hausdorff.

La plupart des méthodes de SLAM effectuent la mise à jour en réalisant une extraction d'amers dans la nouvelle image. Les différentes primitives correspondent à des amers déjà connus, ou bien à de nouvelles primitives qui doivent être rajoutées au modèle. Dans [Davison 98], la recherche de nouveaux amers ne s'effectue que si le nombres de primitives actuellement considérées comme visibles devient trop faible. Dans [Se 02], la pose estimée lors de l'image précédente est utilisée pour projeter les primitives (des points) du modèle sur le plan image courant. La mise à jour s'effectue alors par une recherche dans le voisinage de la position estimée. Le fait que des points étaient déjà visibles dans la vue précédente n'est donc pas utilisée pour mettre à jour leurs positions dans l'image courante. Dans [Ohya 98], les primitives (des lignes) sont aussi projetées en utilisant la pose précédente. Cependant, les contours ne sont pas recherchés dans le voisinage des positions estimées, mais sur toute l'image.

Notons que certaines approches n'effectuent pas de reprojection des amers de la base dans le plan image courant. Citons par exemple les travaux présentés dans [Cobzas 01]. Un système trinoculaire est utilisé afin de créer des images panoramiques, et de récupérer aisément des cartes de profondeurs de l'environnement. Les amers considérés correspondent à des zones planes de l'espace. Pour chaque nouvelle image acquise, une extraction des plans est effectuée. Cette extraction s'effectuant par croissance des régions d'intensité lumineuse équivalente, et segmentation de celles-ci grâce aux cartes de profondeur. La mise en relation avec les amers de la base est alors réalisée par une mesure de corrélation sur les valeurs d'intensité. La position précédente du système robotique dans l'environnement n'est donc pas utilisée pour prévoir où sont censés se trouver les amers.

Dans [Royer 04], le modèle de la scène est constitué d'une séquence de vues associée avec le modèle $3D$, obtenu lors d'une phase préalable de reconstruction. Lors de la navigation, une nouvelle image est traitée pour en extraire tous les points d'intérêt. La détermination des points correspondant à des amers connus est réalisée en appariant l'image courante avec quelques vues de la base. Les vues choisies dépendent de la position estimée du système lors du traitement de la vue précédente. Cepandant, à chaque nouvelle image, le système ne profite pas de la connaissances des amers qui étaient visibles dans la vue précédente.

3.1.3 Méthodes basées sur la connaissance de la trajectoire des primitives

Dans certaines approches, la trajectoire que devront suivre les primitives durant la navigation est directement stockée dans la représentation de l'environnement [Hager 98b, Argyros 01, Gaspar 00, Burschka 01]. Si la route planifiée est bien réalisée, le système robotique connaît approximativement à un instant donné les primitives qui sont susceptibles de rentrer dans le champ de vision de la caméra.

Dans [Rasmussen 96, Hager 98b, Burschka 01], les trajectoires des primitives sont décomposées en séquences. Les transitions entre les séquences correspondent à des apparitions ou des disparitions de primitives dans le champ de vision de la caméra. Dans [Argyros 01], un « cycle de vie » est associée à chaque primitive détectée avec une caméra panoramique. Ce cycle de vie décrit le moment de son apparition et celui de sa disparition. Chaque fin de cycle constitue alors une position que devra atteindre le système robotique durant la navigation.

Cependant, le système robotique ne parvient pas toujours à suivre exactement la trajectoire prédéfinie. Le point d'intérêt peut donc apparaître à une autre position dans l'image. Hager [Hager 98b] effectue cette estimation en calculant les matrices fondamentales entre la vue courante et deux images de la séquence apprise. La position du point dans l'image courante est alors déduite par l'intersection des deux lignes épipolaires respectives à chacune des matrices fondamentales. Le choix des images de référence dans la base n'est pas détaillé, et aucune solution n'est proposée lorsque la matrice fondamentale est dégénérée.

Dans [Argyros 01], la mise à jour des primitives ne s'effectue que lorsque le robot atteint (dans le plan) l'une des positions intermédiaires. L'erreur de localisation vient du fait que le système n'a pas alors nécessairement la même pose que celle associée à l'image de référence. En effet, l'orientation de la caméra panoramique n'est pas contrôlée lors de la navigation. Un décalage horizontal entre la position supposée et la position réelle peut donc être constatée. L'estimation de l'orientation de la caméra, à partir des positions courantes et sauvegardées des primitives déjà visibles permet d'estimer ce décalage, et de localiser les nouveaux amers.

Notons que les trajectoires des primitives ne sont pas toujours connues suite à une phase d'apprentissage. Par exemple dans [Mezouar 02c], ces positions sont estimées par une méthode de planification de trajectoires, à partir d'une série de positions clés que le robot est censé atteindre. Un ensemble d'amers est associé au mouvement entre deux positions clés. La mise à jour des amers visuel n'est effectuée qu'une fois ces positions intermédiaires atteintes.

3.1.4 Positionnement

De manière générale, les approches utilisant des amers locaux sont fondées sur au moins l'une des hypothèses suivantes :

- le modèle tridimensionnel de la scène est connu. La localisation du système par rapport à un repère global permet par reprojection de la scène dans l'espace du capteur de détecter l'apparition de nouveaux amers ;
- la trajectoire des amers durant toute la navigation est connue. Ces trajectoires sont $3D$: à chaque pas de temps est associée une position de ces amers. En particulier, l'instant (et donc les positions) de disparition et d'apparition est une information que connaît le système ;
- le robot converge successivement vers des positions intermédiaires clés. Les amers sont donc associés à un déplacement entre deux positions du chemin. Lorsque le système robotique a atteint une de ces positions intermédiaires, la localisation des nouveaux amers est relativement aisée, puisque l'image fournie par la caméra est quasiment identique à celle associée à la position clé.

La première hypothèse impose de connaître le modèle de la scène, de manière *a priori* ou suite à une phase de reconstruction. La représentation interne de l'environnement de navigation proposée dans ce manuscrit ne contient pas cette information.

La deuxième approche contraint le robot à ne pouvoir se déplaçer qu'en ré-exécutant des trajectoires préalablement apprises. Cette approche est contraignante puisque le robot se voit dans l'incapacité de s'éloigner de cette trajectoire s'il veut pouvoir détecter la rentrée de nouveaux amers dans le champ de vision.

Supposer que le robot parviendra toujours à converger vers chacune des différentes positions intermédiaires est une contrainte forte imposée à la tâche de navigation. Par exemple, l'apparition de nouveaux obstacles dans l'environnement depuis la phase d'apprentissage peut rendre impossible d'atteindre exactement chacune de ces positions clés. Enfin, cette hypothèse (tout comme la précédente) contraint les mouvements possibles du système robotique alors qu'il apparaît logique de lui laisser le plus de liberté d'action afin de pouvoir s'adapter au mieux aux éventuelles évolutions qu'a pu connaître l'environnement de navigation depuis la phase d'apprentissage.

L'approche que nous proposons repose sur l'utilisation du chemin d'images obtenu par la phase de localisation du robot dans l'environnement (présentée dans le chapitre précédent). Les primitives en correspondance entre les différentes vues de ce chemin constituent les amers que nous cherchons à localiser lorsque ceux-ci rentrent dans le champ de vision de la caméra.

Nous ne supposons pas connaître le modèle $3D$ de la scène. La localisation du système robotique revient à situer l'image courante de la caméra par rapport aux différentes vues du chemin, en fonction des primitives en commun.

Le schéma de mise à jour des amers est constitué des étapes suivantes (voir la figure 3.1) :

- *suivi de primitives :* puisque certaines primitives étaient déjà visibles dans l'image précédente de la caméra, celles-ci sont suivies pour connaître leur position dans la nouvelle image acquise. Nous utiliserons l'algorithme KLT pour réaliser ce suivi ;
- *localisation topologique :* la position du système robotique dans l'environnement de navigation n'est pas calculée de manière explicite. La localisation effectuée consiste à déterminer les matrices d'homographie entre la vue courante de la caméra et les différentes vues du chemin avec lesquelles elle partage des primitives ;
- *transfert d'images :* ces matrices permettent ensuite de projeter depuis le chemin vers l'image courante les primitives qui n'étaient pas visibles (ou considérées comme telles) dans la vue précédente de la caméra ;
- *insertion des nouveaux amers :* l'apparition de nouveaux amers est alors constatée si certaines de ces projections sont dans le champ de vision de la caméra.

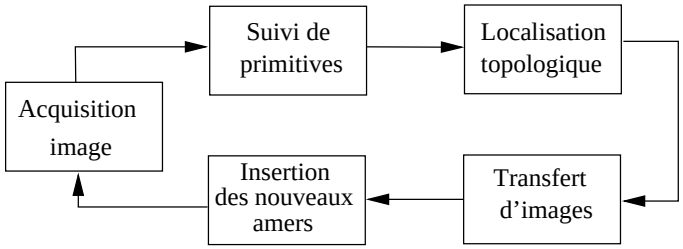


FIG. 3.1: Présentation du schéma de mise à jour des amers

3.2 Gestion de l'évolution des amers visibles

Le cœur de la méthode proposée réside dans le transfert d'images. De manière générale, le transfert d'images consiste à générer une image d'une prise de vue virtuelle à partir de plusieurs vues d'une scène, sans remonter explicitement au modèle tri-dimensionnel de la scène.

Différentes techniques de reprojection ont été proposées dans la littérature. Citons par exemple la triangulation à partir des matrices fondamentales [Hager 97a], ou l'utilisation des tenseurs trifocaux, relations décrivant la géométrie engendrée par trois vues d'une même scène [Shashua 94a, Hartley 94, Avidan 97, Hartley 97b].

Des études réalisées dans le cadre de la reconstruction d'environnements à partir de prises de vue ont montré que l'estimation des tenseurs trifocaux devient instable lorsque les mesures extraites des images sont trop bruitées [Laveau 96]. De plus l'estimation de la matrice fondamentale s'avère plus instable que celle de la matrice d'homographie [Malis 00].

C'est pourquoi nous avons choisi d'effectuer le transfert d'images en utilisant les matrices d'homographie. Ces principes ont déjà été utilisés en robotique, pour des scènes $2D$ et $3D$ [Mezouar 02b], pour planifier dans l'image la trajectoires de primitives pour réaliser des tâches de positionnement.

La méthode de transfert que nous utilisons dans le cas d'une scène tri-dimensionnelle est issue de la méthodologie présentée dans [Shashua 96]. La première section rappelle les principes de cette méthode de transfert. La deuxième section présente plus en détail les différentes étapes permettant de détecter l'apparition de nouveaux amers dans le champ de vision de la caméra. La troisième partie s'intéresse au cas d'une scène plane. Le schéma appliqué repose toujours sur l'estimation et l'utilisation des matrices d'homographie. Comme nous le verrons, la considération d'une scène plane entraîne une simplification notable des opérations à effectuer.

3.2.1 Reprojection à partir de la matrice d'homographie

Comme déjà évoqué dans le chapitre 1, une homographie permet de mettre en relation les projetés de primitives dans deux vues d'une même scène. Soit ψ_1 et ψ_2 ces deux vues. L'homographie ${}^2\mathbf{H}_{p_1}$ est telle que pour tout couple de points appariés $\left({}^1\mathbf{x}_{p_j}, {}^2\mathbf{x}_{p_j}\right)$:

$$ {}^2\mathbf{x}_{p_j} \propto {}^2\mathbf{H}_{p_1}\, {}^1\mathbf{x}_{p_j} + \beta_{1_j}\mathbf{c}_2, \quad \text{avec} \quad \beta_{1_j} = \frac{d_j}{Z_j d_\pi}, \tag{3.1} $$

Z_j est la distance du point $\mathcal{X}_j$ au centre de projection de la caméra associé à la vue ψ_1. d_j correspond à la distance du point au plan de référence. $\mathbf{c}_2$ est la projection du centre de projection de la première caméra sur le plan image de la seconde vue et d_π la distance du centre de projection de ψ_1 au plan π de référence. Le terme β_{i_j} ne dépend que de termes exprimés dans le repère de la première vue.

Une matrice d'homographie estimée à partir d'un ensemble de primitives est définie à un facteur d'échelle près. De ce fait l'équation (3.1) devient :

$$ {}^2\mathbf{x}_{p_j} \propto \alpha\, {}^2\mathbf{H}_{p_1}\, {}^1\mathbf{x}_{p_j} + \beta_{\alpha 1_j}\mathbf{c}_2, \tag{3.2} $$

avec $\beta_{\alpha i_j} = \alpha\beta_{i_j}$. Le facteur d'échelle α rend la parallaxe dépendante du couple d'images à partir duquel elle est déterminée (en utilisant l'équation (1.13)). Ainsi, si l'on considère une troisième

prise de vue ψ_3 de la scène et que l'on calcule l'homographie relative au même plan de référence π entre ψ_1 et ψ_3 :

$$ {}^3\mathbf{x}_{p_j} \propto \alpha'^3\mathbf{H}_{p_1}\,{}^1\mathbf{x}_{p_j} + \beta_{\alpha' 1_j}\mathbf{c}_3, \tag{3.3} $$

et $\beta_{\alpha 1_j} \neq \beta_{\alpha' 1_j}$. Cette inégalité des parallaxes rend impossible, à ce stade, l'estimation de la position d'un point dans la vue ψ_3 à partir de ses projetés dans les vues ψ_1 et ψ_2.

Soit $\mathcal{X}_0$ un point de la scène n'appartenant pas à π. Ses projetés en coordonnées pixelliques dans les vues ψ_1, ψ_2 et ψ_3 sont respectivement ${}^1\mathbf{x}_{p_0}$, ${}^2\mathbf{x}_{p_0}$ et ${}^3\mathbf{x}_{p_0}$. Ils sont tels que :

$$ {}^2\mathbf{x}_{p_0} \propto \alpha^2\mathbf{H}_{p_1}\,{}^1\mathbf{x}_{p_0} + \beta_{\alpha 1_0}\mathbf{c}_2, \quad \text{et} \quad {}^3\mathbf{x}_{p_0} \propto \alpha'^3\mathbf{H}_{p_1}\,{}^1\mathbf{x}_{p_0} + \beta_{\alpha' 1_0}\mathbf{c}_3 \tag{3.4} $$

Les relations ci-dessus étant définies à un facteur d'échelle près, elles restent valident si on les multiplie par un scalaire. Shashua propose de mettre les deux homographies à l´échelle de sorte que [Shashua 96] :

$$ {}^2\mathbf{x}_{p_0} \propto {}^2\mathbf{H}'_{p_1}\,{}^1\mathbf{x}_{p_0} + \mathbf{c}_2, \quad \text{et} \quad {}^3\mathbf{x}_{p_0} \propto {}^3\mathbf{H}'_{p_1}\,{}^1\mathbf{x}_{p_0} + \mathbf{c}_3, \tag{3.5} $$

où :

$$ {}^2\mathbf{H}'_{p_1} = \frac{\alpha}{\beta_{\alpha 1_0}}\,{}^2\mathbf{H}_{p_1} \quad \text{et} \quad {}^3\mathbf{H}'_{p_1} = \frac{\alpha'}{\beta_{\alpha' 1_0}}\,{}^3\mathbf{H}_{p_1} $$

L'utilisation de ces matrices d'homographies dans les équations (3.1) et (3.3) revient à diviser la parallaxe de chacun des points par la parallaxe associée au point $\mathcal{X}_0$. Pour un point $\mathcal{X}_j \notin \pi$, la parallaxe relative à la matrice d'homographie entre les images ψ_1 et ψ_2 devient :

$$ \beta'_{1_j} = \frac{\beta_{\alpha 1_j}}{\beta_{\alpha 1_0}} = \frac{\alpha d_j}{Z_j d_\pi}\frac{Z_0 d_\pi}{\alpha d_0} = \frac{d_j Z_0}{d_0 Z_j} $$

La parallaxe mise à l'échelle β'_{1_j} ne dépend plus du facteur d'échelle α. Elle peut donc être utilisée pour projeter le point $\mathcal{X}_j$ dans la vue ψ_3, à partir de ses coordonnées dans les images ψ_1 et ψ_2 et de sa parallaxe associée qui peut être estimée entre ces deux vues.

Par cette opération, Shashua montre que [Shashua 96] :

- **si** l'on connaît deux homographies ${}^2\mathbf{H}_{p_1}$ et ${}^3\mathbf{H}_{p_1}$, respectivement entre les couples d'images (ψ_1, ψ_2) et (ψ_1, ψ_3), relatives au même plan de référence π,
- **si** les deux homographies sont mises à l'échelle par rapport à un point $\mathcal{X}_0 \notin \pi$ de la scène, de sorte que ${}^2\mathbf{x}_{p_0} \propto {}^2\mathbf{H}_{p_1}\,{}^1\mathbf{x}_{p_0} + \mathbf{c}_2$ et ${}^3\mathbf{x}_{p_0} \propto {}^3\mathbf{H}_{p_1}\,{}^1\mathbf{x}_{p_0} + \mathbf{c}_3$,
- **alors** la parallaxe d'un autre point de la scène est identique pour les deux couples de vue.

Notons qu'il n'est pas nécessaire de garder toujours le même point de référence $\mathcal{X}_0$ pour mettre à l'échelle l'homographie et les parallaxes. Si l'on considère deux parallaxes β_1 et β_2 associées respectivement aux points $\mathcal{X}_1$ et $\mathcal{X}_2$, et que celles-ci sont mises à l'échelle par rapport à β_0, nous avons (avec une vue ψ_i de référence) :

$$ \beta'_{i_0} = 1, \quad \beta'_{i_1} = \frac{Z_0}{Z_1}\frac{d_1}{d_0}, \quad \beta'_{i_2} = \frac{Z_0}{Z_2}\frac{d_2}{d_0} $$

Si l'on décide maintenant de mettre à l'échelle ces parallaxes par rapport à β'_{i_2}, on obtient :

$$ \beta'_{i_0} = \frac{Z_2}{Z_0}\frac{d_0}{d_2}, \quad \beta'_{i_1} = \frac{Z_0}{Z_1}\frac{d_1}{d_0}\frac{Z_2}{Z_0}\frac{d_0}{d_2} = \frac{Z_2}{Z_1}\frac{d_1}{d_2}, \quad \beta'_{i_2} = 1, $$

où l'on note que les parallaxes obtenues sont indépendantes de la première mise à l'échelle.

Cette propriété va être utilisée pour réaliser le transfert d'images, comme l'illustre la figure 3.2. Les correspondances entre deux vues du chemin ψ_i et ψ_{i+1} permettent d'estimer la géométrie épipolaire associée, et plus particulièrement les parallaxes relatives à la vue ψ_i et à un plan π de la scène (étape 1). Si l'on suppose que, durant le mouvement, la vue ψ_t fournie par la caméra embarquée dispose d'assez d'amers en commun avec la vue ψ_i du chemin, la matrice d'homographie associée peut être estimée (étape 2). Grâce aux parallaxes calculées précédemment, à la matrice d'homographie entre ψ_i et ψ_t, et en choisissant un point de référence commun à ψ_i et ψ_t, il est possible de reprojeter des amers de ψ_i et ψ_{i+1} dans l'image courante (étape 3). Les nouveaux points rentrant dans le champ de vision de la caméra peuvent ainsi être détectés.

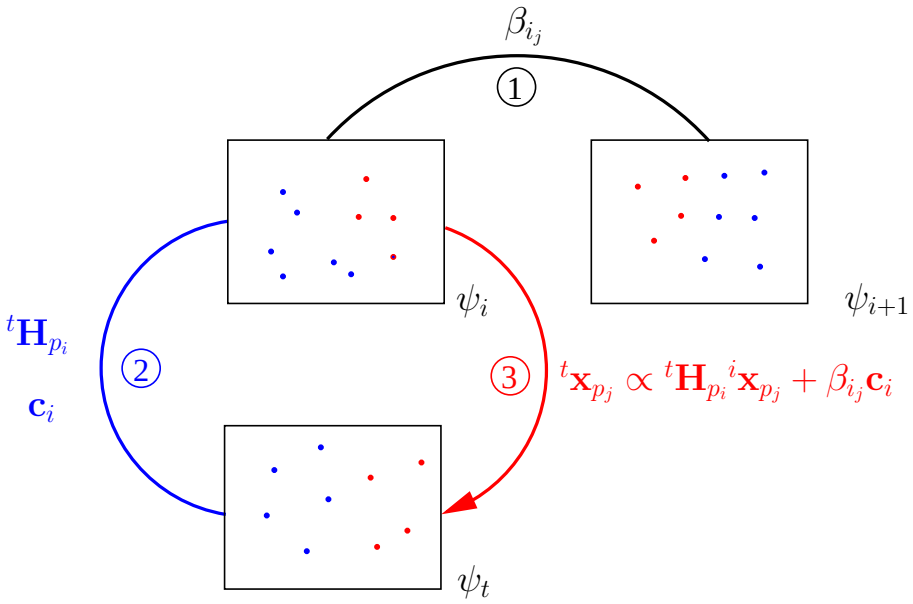


FIG. 3.2: Mise à jour des amers visibles par transfert d'images. 1 : calcul des parallaxes entre deux vues successives ψ_i et ψ_{i+1} du chemin, 2 : estimation de l'homographie entre la vue courante de la caméra ψ_t et la vue ψ_i du chemin, 3 : projection des points appariés entre ψ_i et ψ_{i+1} dans la vue courante. L'étape 1 n'a besoin d'être réalisée qu'une seule fois, lors de la création de la base d'images. Les points bleus sont des points initialement appariés entre ψ_i et ψ_{i+1}, et qui sont déjà connus comme visibles dans l'image courante. Les points rouges sont des points initialement appariés entre ψ_i et ψ_{i+1}, et qui n'étaient pas considérés comme visibles jusqu'à présent. Le transfert d'images permet de détecter leur apparition dans le champ de vision de la caméra.

La section suivante présente comment ce transfert peut être utilisé de manière générique et automatique pour assurer la mise à jour des primitives visibles, dans le cas d'une scène tridimensionnelle.

3.2.2 Approche proposée : cas d'une scène tri-dimensionnelle

La section 3.2.2.1, après un rappel sur les notations utilisées, présente et décrit l'organigramme des opérations permettant de réaliser la gestion de la mise à jour des amers visibles. Les deux sections suivantes sont des approfondissements sur le fonctionnement de deux opérations critiques de cette boucle de traitement. La section 3.2.2.2 détaille les informations qui sont nécessaires pour définir une épipole à partir de la décomposition d'une matrice d'homographie.

3.2.2.1 Protocole de mise à jour des primitives visibles

La figure 3.3 illustre les différentes notations qui vont être utilisées dans cette partie.

Le chemin d'images est décrit par un ensemble d'amers. Ceux-ci correspondent aux points qui sont mis en correspondance entre deux vues successives. On note $\mathcal{M}_i$ l'ensemble de points appariés entre les images ψ_i et ψ_{i+1} du chemin.

Durant la navigation, les points d'intérêt visibles dans l'image fournie par la caméra peuvent être mis en relation avec les images du chemin dans lesquelles ils ont été initialement définis. On note $\mathcal{V}_i$ l'ensemble des points définis dans la vue ψ_i qui sont visibles dans l'image courante ψ_t. Un point de l'image courante peut être associé à plusieurs images du chemin (au minimum 2 puisqu'on ne considère que les points mis en correspondance entre deux images successives du chemin).

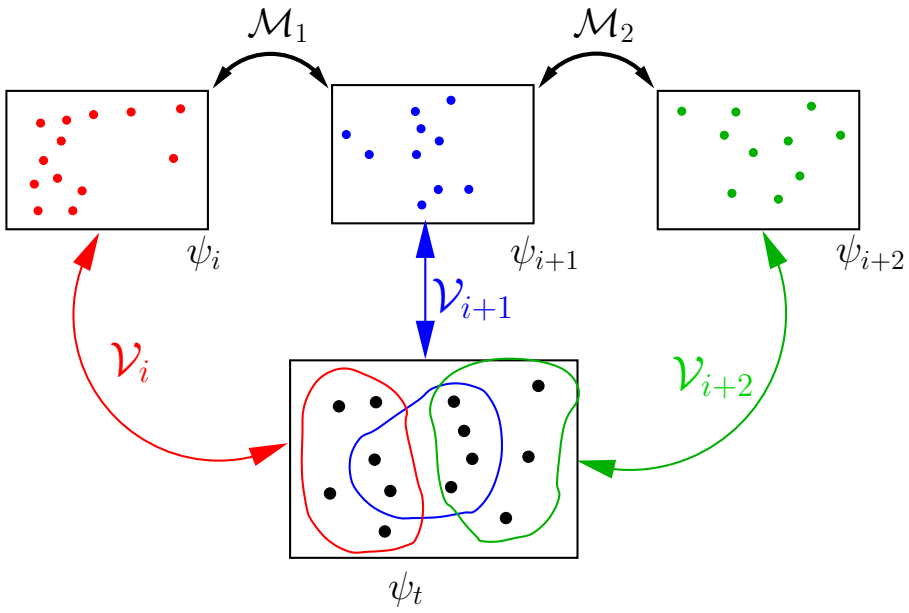


FIG. 3.3: Notations utilisées : $\mathcal{V}_i$ est l'ensemble des primitives de l'image ψ_i actuellement visibles dans l'image fournie par la caméra. $\mathcal{M}_i$ correspond aux points appariés entre les vues ψ_i et ψ_{i+1}.

La figure 3.4 résume les différentes étapes permettant de détecter l'apparition de nouvelles primitives. Le suivi des points entre l'image ψ_{t-1} et ψ_t est réalisé par la méthode KLT [Shi 94] (dont les principes ont été présentés dans la section 1.2.4). La localisation topologique s'effectue en estimant les matrices d'homographie entre la vue ψ_t et les différentes images du chemin.

Les termes de parallaxes calculés sur le chemin d'images permettent alors de projeter sur le plan image courant les points qui n'étaient pas considérés comme visibles. L'apparition de nouvelles primitives est alors constatée si elles se projettent dans le cadre de l'image.

Chacune des étapes de ce schéma est détaillée par la suite. Nous présenterons de plus les informations qui doivent être calculées au lancement de la boucle de mise à jour.

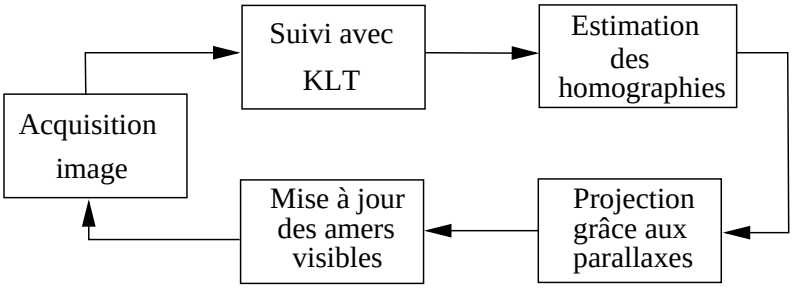


FIG. 3.4: Schéma de mise à jour des primitives visibles

Nous supposons pour l'instant que la phase d'initialisation a permis de définir pour chaque couple d'images successives (ψ_i, ψ_{i+1}) du chemin la matrice d'homographie associée ${}^{i+1}\mathbf{H}_{p_i}$. De même, les termes de parallaxes β_{i_j} ont pu être estimés pour chacune des correspondances entre ces deux vues.

Le traitement de la vue ψ_{t-1} fournit les coordonnées d'un ensemble de points visibles ${}^{t-1}\mathbf{x}_{p_j}$. Les ensembles $\mathcal{V}_i$ permettent de faire le lien entre les points visibles et les images du chemin dont ils sont issus.

Suivi classique de points

Une nouvelle image ψ_t est acquise par la caméra. L'utilisation du suivi de KLT permet de mettre à jour la position des primitives, soit ${}^{t}\mathbf{x}_{p_j}$. Si le suivi d'un point a échoué (l'erreur résiduelle ε obtenue avec l'équation (1.21) est trop importante), les ensembles $\mathcal{V}_i$ sont mis à jour en conséquence.

Dans le cadre de tâches robotiques de positionnement ou de navigation, les mouvements de la caméra peuvent être supposés continus. De ce fait, le mouvement apparent des primitives dans l'image évolue de manière relativement homogène. Pour accélérer la phase de minimisation de ε, la mesure de translation image d d'un point est initialisée comme étant la translation obtenue pour cette primitive lors du traitement de l'image précédente ψ_{t-1}.

Connaissant la position dans ψ_t d'un ensemble de primitives, il est alors possible de situer la caméra par rapport au chemin d'images, en estimant les matrices d'homographie vers ces différentes vues, comme cela va être décrit à présent.

Mise à jour des homographies

L'étape suivante consiste à déterminer les matrices d'homographies entre la vue courante ψ_t et les différentes images ψ_i du chemin. Nous ne présentons que l'estimation entre les vues ψ_i et ψ_t. Le principe est le même pour les autres images du chemin.

Les termes de parallaxes restent identiques tant que l'on conserve la même vue et le même plan de référence. L'estimation de l'homographie s'effectue en respectant ces deux contraintes.

L'homographie est déterminée en utilisant la méthode de Malis [Malis 00]. Dans cette approche, le plan de référence est défini par trois points. La mise à jour de l'homographie ne peut donc s'effectuer que si les trois points ayant servi à définir le plan de référence de l'homographie ${}^{i+1}\mathbf{H}_{p_i}$ sont actuellement visibles. Cette information est obtenue en vérifiant si ces points appartiennent à l'ensemble $\mathcal{V}_i$.

De plus, l'estimation d'une homographie nécessite au moins huit correspondances de points. La taille de l'ensemble $\mathcal{V}_i$, soit $\text{card}(\mathcal{V}_i)$ permet de s'assurer que cette contrainte est respectée.

Si assez de points sont visibles, et si les trois points définissant le plan de la scène le sont aussi, la matrice d'homographie ${}^{t}\mathbf{H}_{p_i}$ est mise à jour. À partir de cette homographie, il est possible de déterminer l'épipole $\mathbf{c}_t$, projection dans l'image courante ψ_t du centre de projection de la vue ψ_i (la section 3.2.2.2 décrit plus en détail l'estimation de cette épipole).

Si les deux conditions données ci-dessus ne sont pas respectées, l'homographie ${}^{t}\mathbf{H}_{p_i}$ n'est pas mise à jour, et ne pourra donc pas être utilisée pour effectuer des reprojections.

Connaissant la matrice d'homographie et l'épipole, il est possible de projeter sur le plan image courant les primitives de l'ensemble $\mathcal{M}_i$ qui ne sont pas actuellement considérées comme visibles.

Projection des primitives du chemin

La phase de projection ne concerne que les ensembles de primitives $\mathcal{M}_i$ dont l'homographie associée ${}^{t}\mathbf{H}_{p_i}$ a pu être mise à jour.

Parmi tous les points visibles de l'ensemble $\mathcal{M}_i$, un point n'appartenant pas au plan de référence est choisi (un tel point possède un terme de parallaxe non nul). La parallaxe β_{t_j} associée à la matrice ${}^{t}\mathbf{H}_{p_i}$ est calculée en utilisant l'équation (1.13) page 19, que nous rappelons ici :

$$\beta_{t_j} = -\frac{\left({}^{t}\mathbf{H}_{i_j}\,{}^{i}\mathbf{x}_{p_j} \wedge {}^{t}\mathbf{x}_{p_j}\right)^{\top}\left(\mathbf{c}_t \wedge {}^{t}\mathbf{x}_{p_j}\right)}{\|\mathbf{c}_t \wedge {}^{t}\mathbf{x}_{p_j}\|^2}, \tag{3.6}$$

où $\mathbf{c}_t$ dépend de l'homographie $\mathbf{H}_{p_i}$ considérée. La matrice d'homographie ${}^{t}\mathbf{H}_{p_i}$ peut alors être mise à l'échelle :

$${}^{t}\mathbf{H'}_{p_i} = \frac{1}{\beta_{t_j}}{}^{t}\mathbf{H}_{p_i}$$

Conformément à la technique présentée dans la section 3.2.1, les parallaxes des points de $\mathcal{M}_i$ estimées entre les vues ψ_i et ψ_{i+1} sont normalisées par rapport à celle associée au point de référence choisi. Si l'on note ${}^{i}\mathbf{x}_{p_k}$ un point actuellement non visible, on obtient :

$$\beta'_{i_k} = \frac{\beta_{i_k}}{\beta_{i_j}}$$

La position de ce point sur le plan image courant peut ainsi être mise à jour suivant l'équation :

$$ {}^{t}\mathbf{x}_{p_k} \propto {}^{t}\mathbf{H'}_{p_i}\,{}^{i}\mathbf{x}_{p_k} + \beta'_{i_k}\mathbf{c}_t \tag{3.7} $$

Mise à jour des points visibles

La phase précédente fournit les coordonnées sur le plan image courant des primitives qui n'étaient pas considérées comme visibles dans l'image précédente. Celles qui se projettent à l'intérieur du champ de vision de la caméra (délimité par la taille de l'image) sont donc potentiellement visibles.

En effet, si l'équation (3.7) de reprojection est géométriquement exacte, il s'avère en pratique que les coordonnées ainsi obtenues peuvent ne pas correspondre à la projection réelle du point dans l'image courante.

La méthode de suivi KLT correspond à une corrélation différentielle. Cela implique que la détermination des coordonnées d'un point ${}^{t}\mathbf{x}_{p_j}$ dans une nouvelle image s'effectue en supposant que la position ${}^{t-1}\mathbf{x}_{p_j}$ de cette primitive dans l'image précédente est exacte. Si ce n'est pas le cas, le point suivi peut différer du point d'intérêt. Il est donc primordial lors du rajout d'un nouveau point de vérifier la validité de sa position estimée, afin de s'assurer que le point qui va être suivi est bien le bon. Ce contrôle est réalisé en estimant le modèle affine permettant de mettre en correspondance une fenêtre de ψ_t autour de la position supposée du point avec une fenêtre de référence associée à ce point et définie dans une des images du chemin où il a été initialement détecté.

On considère ici que le point ${}^{t}\mathbf{x}_{p_j}$ traité provient d'un transfert d'images depuis les vues ψ_i et ψ_{i+1}. L'estimation du modèle affine s'effectue comme présentée dans la partie 1.2.4. La qualité de la position obtenue par reprojection est associée au résidu ε défini comme étant :

$$ \varepsilon = \int\int_{\mathcal{W}} [\psi_t(\mathbf{A}\,{}^{t}\mathbf{x}_{p_j\mathcal{W}}) - \psi_i({}^{i}\mathbf{x}_{p_j\mathcal{W}})]^2 w(\mathbf{x}_{\mathcal{W}}) d\mathbf{x}, \tag{3.8} $$

où ${}^{i}\mathbf{x}_{p_j}$ est le projeté du point $3D$ $\mathcal{X}_j$ considéré dans la vue ψ_i du chemin, et $\mathbf{A}$ la transformation affine permettant de recaler la fenêtre courante sur la fenêtre de référence. La projection estimée est considérée comme correcte si le résidu est inférieur à un seuil donné ε_t.

De manière pratique, si la projection n'est pas correcte, la minimisation réalisée pour déterminer le modèle affine $\mathbf{A}$ diverge très rapidement. Quelques itérations sont donc suffisantes pour détecter que la position estimée n'est pas valide.

Puisque le point considéré a été initialement défini dans les vues ψ_i et ψ_{i+1} du chemin, la référence peut être choisie dans chacune de ces deux images. Si la détermination du modèle affine depuis la vue ψ_i n'a pas convergé, la fenêtre définie dans l'image ψ_{i+1} est considérée comme la nouvelle référence, et une nouvelle tentative est effectuée.

Si aucune des images de référence ne permet d'obtenir un résidu satisfaisant, on suppose alors que le point réel se situe dans un voisinage proche de la position supposée par reprojection. Les points d'intérêts sont extraits dans une fenêtre centrée autour de la position estimée. Pour chacun des points trouvés, le modèle affine est estimé. Le point fournissant le plus faible résidu est alors considéré comme la position exacte du point. Dans le cas où aucun candidat n'est trouvé, ou si

le calcul du modèle affine échoue pour tous, le point n'est pas considéré comme visible (il n'est donc pas rajouté à $\mathcal{V}_i$).

Cette opération conclut la gestion de la mise à jour des primitives visibles. Le système est alors prêt à traiter une nouvelle prise de vue ψ_{t+1}. La partie suivante présente plus en détail les contraintes liées à l'estimation des épipoles. Nous en profiterons pour définir plus précisément les informations qui doivent être calculées avant le lancement de la tâche de suivi, afin d'assurer le bon fonctionnement de la mise à jour.

3.2.2.2 Estimation de l'épipole et initialisation de la boucle

L'épipole $\mathbf{c}_t$ est obtenue en décomposant la matrice d'homographie ${}^t\mathbf{H}_{n_i}$ (cette matrice d'homographie étant déduite de ${}^t\mathbf{H}_{p_i}$ et des paramètres de la caméra, en utilisant (1.10) page 18). Comme il a déjà été mentionné dans la section 1.1.3.4, plusieurs décompositions sont possibles à partir d'une même matrice d'homographie [Faugeras 88]. Dans [Malis 00], cette ambiguïté est levée en s'appuyant sur le fait que la position désirée du plan image est quasiment parallèle au plan de référence. Cette hypothèse permet de donner une estimation de la normale désirée, et la décomposition donnant la normale la plus proche de cette estimée est alors choisie.

Il apparaît difficile d'appliquer une telle méthode dans notre contexte, puisqu'aucune connaissance *a priori* n'est connue sur la scène décrite par la base d'images. Définir ces normales manuellement n'est pas non plus une solution envisageable.

Cette ambiguïté doit pourtant être levée non seulement pour estimer les épipoles durant le déplacement, mais aussi pour l'initialisation du système, afin d'estimer les parallaxes des points du chemin. La résolution du problème à l'initialisation permet de choisir les bonnes décompositions des matrices ${}^t\mathbf{H}_{p_i}$ durant le mouvement.

Pour ce faire, les images successives du chemin sont traitées par ensemble de trois. Ainsi, si l'on considère les vues ψ_{i-1}, ψ_i et ψ_{i+1}, les ensembles de correspondances $\mathcal{M}_{i-1}$ et $\mathcal{M}_i$ associées permettent de définir deux homographies ${}^{i-1}\mathbf{H}_{p_i}$ et ${}^{i+1}\mathbf{H}_{p_i}$, où l'image de référence est ψ_i dans les deux cas. Le plan de référence doit être le même pour ces deux matrices. Il est donc nécessaire de choisir les même points de référence pour l'estimation des deux homographies. La sélection automatique des points décrivant le plan de référence est détaillée dans l'annexe A.

Les normales issues des décompositions des deux homographies sont exprimées dans le repère associé à la vue ψ_i. Les normales doivent donc être identiques dans les deux décompositions. Les décompositions choisies des deux homographies sont donc celles dont les normales associées ${}^i\mathbf{n}_1$ et ${}^i\mathbf{n}_2$ forment entre elles l'angle le plus faible.

Il est alors possible de définir à partir de l'épipole obtenue les termes de parallaxe de l'ensemble $\mathcal{M}_i$ qui seront utilisés pour effectuer la mise à jour durant la navigation.

La matrice d'homographie estimée en ligne ${}^t\mathbf{H}_{p_i}$ a les mêmes vue et plan de référence que la matrice ${}^{i+1}\mathbf{H}_{p_i}$ estimée sur le chemin d'images. Dans les deux cas, la normale au plan de référence doit donc être identique. La bonne décomposition de l'homographie ${}^t\mathbf{H}_{p_i}$ est donc choisie comme étant celle fournissant la normale la plus proche de celle estimée hors-ligne. De cette décomposition est déduite l'épipole et la reprojection des primitives de l'ensemble $\mathcal{M}_i$ sur le plan image courant peut alors être réalisée.

En conclusion, un ensemble d'informations sont nécessaires au bon déroulement de la mise à jour des amers visibles. Ces informations sont calculées directement sur le chemin d'images, avant le début des mouvements du système robotique. Pour chaque ensemble d'appariement $\mathcal{M}_i$, il est déterminé et conservé :

- les indices des trois points définissant le plan de référence,
- la normale ${}^{i}\mathbf{n}$ associée à ce plan, exprimée dans le repère associé à la vue ψ_i,
- les parallaxes de chacun des points de $\mathcal{M}_i$, obtenues par l'équation (3.6).

3.2.3 Cas d'une scène plane

Le protocole présenté ci-dessus permet de gérer des environnements de navigation tridimensionnels. Si l'on considère une scène plane, le problème de la mise à jour des points visibles devient beaucoup plus simple.

Nous présentons tout d'abord les relations de transfert d'images dans le cas d'une scène plane. La boucle de mise à jour reste dans les grandes lignes identique à celle présentée sur le schéma 3.4. Nous ne détaillerons ensuite que les étapes dont les traitements diffèrent de ceux présentés dans le cas d'une scène $3D$.

3.2.3.1 Transfert de primitives

Dans le cas d'une scène plane, le terme de parallaxe apparaissant dans la relation (3.1) est nul (puisque la distance des points au plan de référence est nulle). La relation liant les projetés entre deux prises de vue se réduit donc à :

$$ {}^{2}\mathbf{x}_{p_j} \propto {}^{2}\mathbf{H}_{p_1}\,{}^{1}\mathbf{x}_{p_j} \tag{3.9} $$

De la même manière, si des primitives de l'image ψ_2 sont en correspondance avec celles d'une troisième vue de la scène, une relation analogue peut être définie :

$$ {}^{3}\mathbf{x}_{p_j} \propto {}^{3}\mathbf{H}_{p_2}\,{}^{2}\mathbf{x}_{p_j} \tag{3.10} $$

Le produit matriciel de deux matrices de transformation projective est une matrice de transformation projective. Ce produit permet de mettre en relation les primitives des vues ψ_1 et ψ_3 :

$$ {}^{3}\mathbf{x}_{p_j} \propto {}^{3}\mathbf{H}_{p_2}\,{}^{2}\mathbf{H}_{p_1}\,{}^{1}\mathbf{x}_{p_j} \propto {}^{3}\mathbf{H}_{p_1}\,{}^{1}\mathbf{x}_{p_j} \tag{3.11} $$

En généralisant cette composition, il est possible de déterminer la matrice de transformation projective entre deux points de vue d'un même plan, même si les deux images associées ne contiennent pas de primitives en correspondance :

$$ {}^{j}\mathbf{H}_{p_k} \propto \prod_{i=k}^{j-1} {}^{i+1}\mathbf{H}_{p_i} $$

Cette dernière relation est celle utilisée pour projeter des points du chemin d'image dans l'image courante de la caméra durant son mouvement. La section suivante présente les différents traitements permettant de gérer la mise à jour des primitives visibles dans le cas d'une scène plane.

3.2.3.2 Schéma de mise à jour des primitives

Les traitements qui diffèrent de ceux appliqués dans le cas $3D$ concernent la mise à jour des homographies, ainsi que la phase de transfert des primitives du chemin d'images vers la prise de vue courante. Nous nous limiterons à la présentation de ces deux opérations.

Nous supposons ici que les homographies ${}^{i+1}\mathbf{H}_{p_i}$ relatives aux couples d'images successives du chemin ont été déterminées, en résolvant le système suivant :

$$ {}^{i+1}\mathbf{x}_{p_j} \propto {}^{i+1}\mathbf{H}_{p_i}\, {}^{i}\mathbf{x}_{p_j} \text{ , } \forall\ \mathcal{X}_j \in M_i $$

Mise à jour des homographies

Comme dans le cas $3D$, l'étape de mise à jour des homographies est précédée du suivi des primitives visibles dans l'image précédente ψ_{t-1}, afin de connaître leur position dans l'image fraîchement acquise ψ_t.

Le calcul d'une homographie dans le cas d'une scène plane nécessite au moins la connaissance de quatre primitives en commun entre les deux vues considérées. Pour toutes le vues dont la taille de l'ensemble $\mathcal{V}_i$ associé est supérieure à ce seuil, les matrices d'homographies ${}^{t}\mathbf{H}_{p_i}$ les reliant à la prise de vue courante peuvent donc être estimées en résolvant le système :

$$ {}^{t}\mathbf{x}_{p_j} \propto {}^{t}\mathbf{H}_{p_i}\, {}^{i}\mathbf{x}_{p_j} \text{ , } \forall\ \mathcal{X}_j \in V_i $$

La composition des matrices d'homographies du chemin permet de traiter les prises de vue ne possédant pas (ou pas assez) de primitives en commun avec l'image courante pour pouvoir résoudre le système ci-dessus. Par exemple, si l'on considère que la vue ψ_l du chemin est dans ce cas, et que la matrice d'homographie ${}^{t}\mathbf{H}_{p_i}$ (avec $i < l$) a pu être estimée à partir des primitives en correspondances entre ψ_i et ψ_t, alors ${}^{t}\mathbf{H}_{p_l}$ est définie comme étant :

$$ {}^{t}\mathbf{H}_{p_l} \propto {}^{t}\mathbf{H}_{p_i}\, {}^{i}\mathbf{H}_{p_l}, $$

où ${}^{i}\mathbf{H}_{p_l}$ est estimée par composition des matrices d'homographie calculées sur le chemin d'images.

Connaissant les matrices d'homographie reliant la vue courante à chacune des vues du chemin, nous montrons maintenant comment s'effectue la mise à jour des primitives.

Mise à jour des primitives visibles

La projection ne concerne que les primitives des ensembles $\mathcal{M}_i$ qui n'étaient pas visibles dans l'image précédente de la caméra ψ_t. Leur position sur le plan image courant est définie comme étant :

$$ {}^{t}\mathbf{x}_{p_j} \propto {}^{t}\mathbf{H}_{p_i}\, {}^{i}\mathbf{x}_{p_j} \quad \text{si} \ \ \mathcal{X}_j \in \mathcal{M}_i $$

Notons que la composition des matrices d'homographie permet potentiellement de projeter toutes les primitives du chemin d'images sur le plan image courant. Le nombre d'ensembles $\mathcal{M}_i$ concernés par cette reprojection peut être limité en ne considérant que ceux qui sont susceptibles de voir prochainement certains de leurs points rentrer dans le champ de vision de la caméra (par exemple, si seuls les deux ensembles $\mathcal{M}_2$ et $\mathcal{M}_3$ possèdent des primitives

visibles dans l'image précédente, la projection peut être effectuée pour ces deux ensembles, ainsi que celui définissant la prochaine zone que le robot est susceptible de traverser, à savoir $\mathcal{M}_4$).

Cette reprojection permet alors de détecter les primitives se projetant à l'intérieur de la zone délimitée par le cadre de l'image. Un contrôle dans l'image courante permet alors de confirmer (ou d'infirmer) cette apparition (les même techniques que celles présentées dans le cas $3D$ sont pour cela utilisées). Le système est alors prêt à traiter une nouvelle prise de vue.

La partie suivante va montrer des résultats expérimentaux, pour des environnements plans et des environnements $3D$.

3.2.4 Résultats expérimentaux

3.2.4.1 Cas d'une scène plane

Les figures 3.5 et 3.6 présentent les ensembles d'appariement constituant la description d'un environnement plan. Les images initiale et désirée ne possèdent aucune primitive en commun.

Des images acquises durant les mouvements de la caméra sont présentées sur les figures 3.7 et 3.8. Les mouvements du robot sont contrôlés suivant le formalisme proposé dans le chapitre suivant. La dernière image de la figure 3.6 correspond à la position que doit atteindre le système robotique.

Sur les quatre dernières images, les points verts correspondent aux primitives du dernier ensemble $\mathcal{M}_5$. Ces primitives doivent atteindre les positions qui figurent en bleu sur les dernières images. La plupart des points sont proches de leur position désirée. Notons que quelques erreurs de localisation persistent. Cependant, le fait que la majorité des points est correctement située minimise l'influence de ces mauvais points lors de l'estimation des homographies (et aussi lors de l'estimation de la loi de commande qui est présentée dans le chapitre suivant).

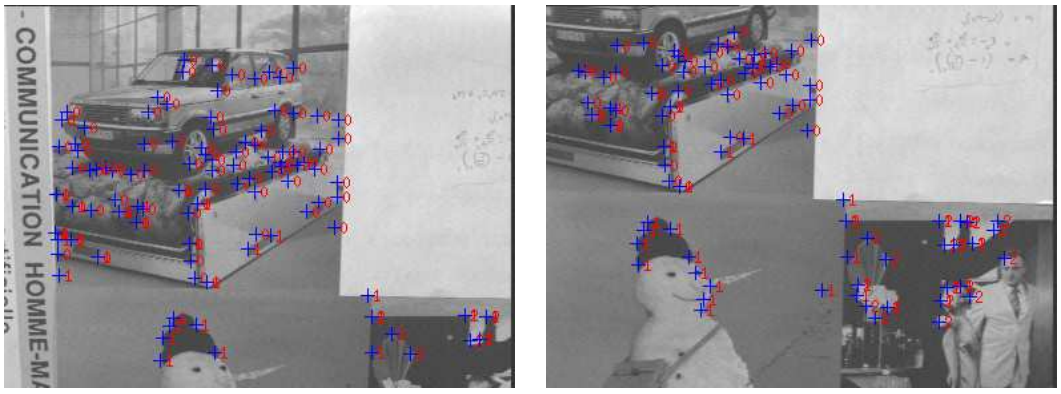
Notons de plus que le système robotique n'atteint pas les positions associées aux images du chemin, ce qui n'empêche pas le système de mettre à jour les primitives visibles.

FIG. 3.5: Chemin d'images (ensembles de $\mathcal{M}_0$ à $\mathcal{M}_3$). La première vue correspond à la position initiale.

$\mathcal{M}_4$

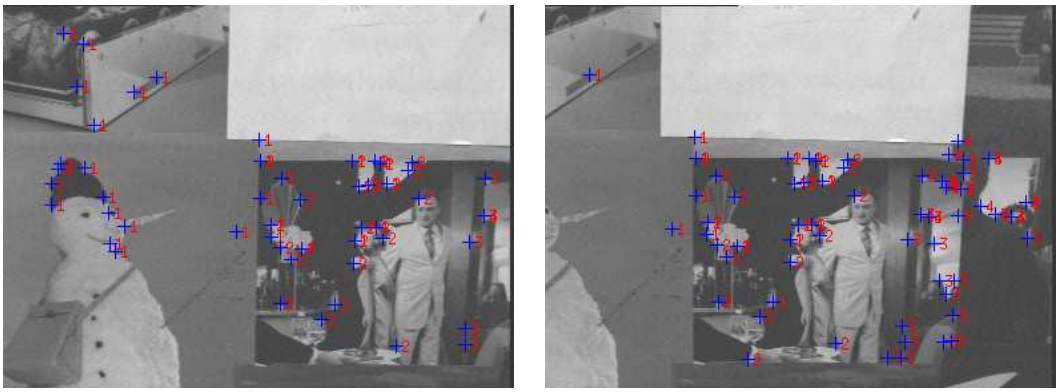
$\mathcal{M}_5$

FIG. 3.6: Suite du chemin d'images (ensembles de $\mathcal{M}_4$ à $\mathcal{M}_5$). La dernière image correspond à la position que doit atteindre le système robotique

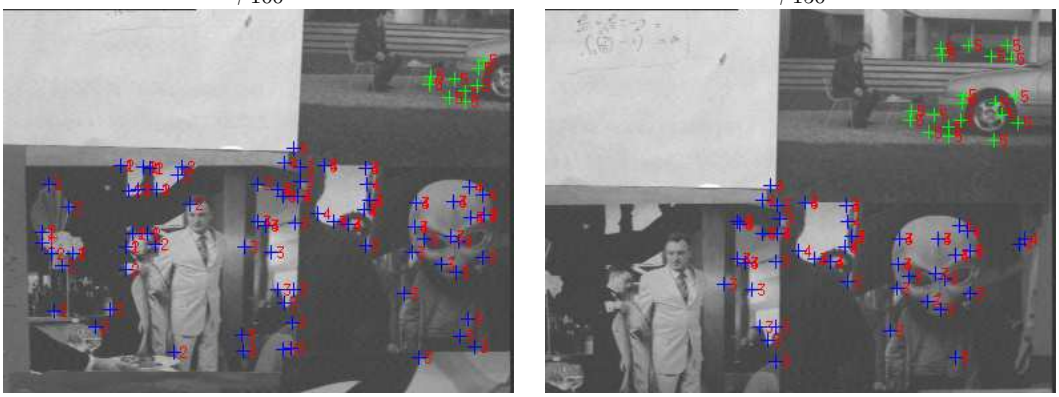


ψ_0 ψ_{50}

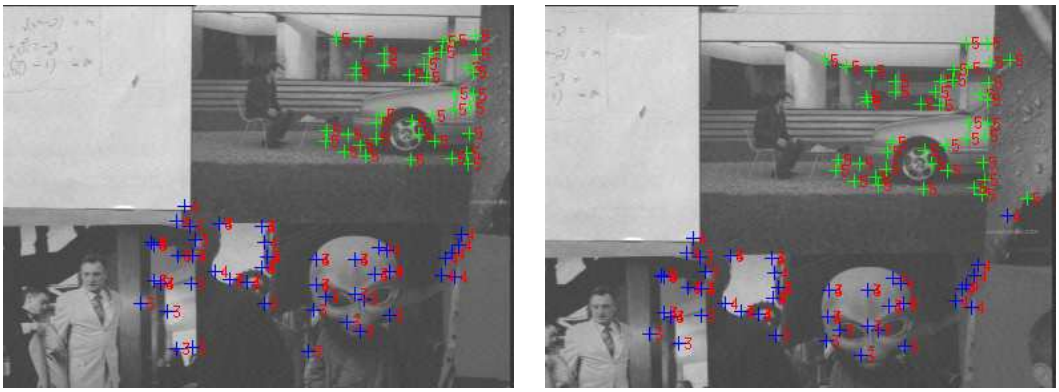
FIG. 3.7: Visualisation de prises de vue durant le mouvement. Les points bleus sont les points suivis. À chaque point est associé un ensemble d'intérêt (indiqué en rouge ici).



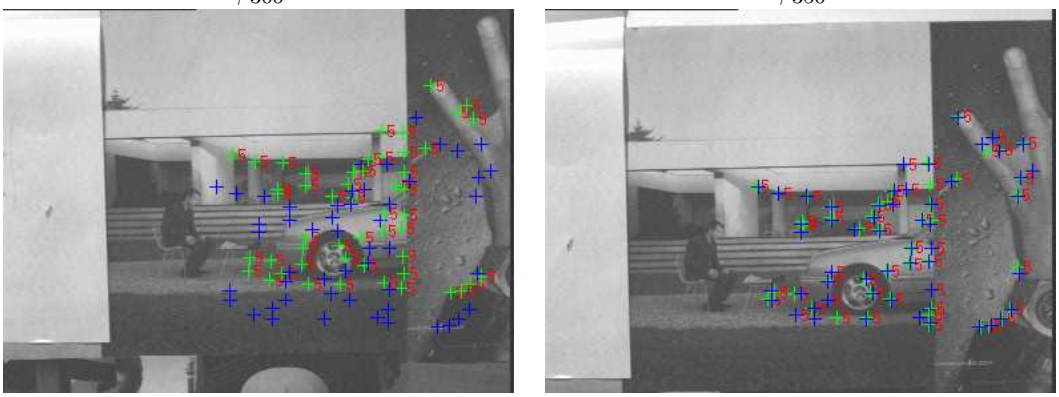
ψ_{100} ψ_{150}



ψ_{200} ψ_{250}



ψ_{300} ψ_{350}



ψ_{400} ψ_{500}

FIG. 3.8: Suite de la visualisation. Dans les deux dernières vues, les points bleus montrent les positions dans l'image que doivent atteindre les points du dernier ensemble.

3.2.4.2 Cas d'une scène tridimensionnelle

Les expériences ci-dessous s'intéressent à des scènes $3D$. La première expérience concerne un environnement intérieur. Les ensembles de correspondances $\mathcal{M}_i$ sont présentés sur les figures 3.9 et 3.10. Bien que l'image initiale et l'image finale partagent des projections de mêmes primitives $3D$ de la scène, le changement d'échelle entre ces deux vues rend impossible la détection des correspondances entre ces deux vues (avec l'algorithme d'appariement utilisé). Le déplacement considéré est principalement réalisé suivant l'axe optique.

La figure 3.11 présente les résultats de suivi et de mise à jour obtenus alors que la caméra se déplace dans l'environnement. Les primitives bleus correspondent aux primitives qui sont visibles depuis la première image. Les primitives rouges sont les primitives qui ont été localisées par notre méthode de mise à jour. Les numéros à côté des points correspondent aux ensembles $\mathcal{M}_i$ auxquels ceux-ci appartiennent.

La première constatation qui peut être effectuée est que toutes les primitives visibles dans la dernière image ont été localisées par la technique de mise à jour que nous proposons.

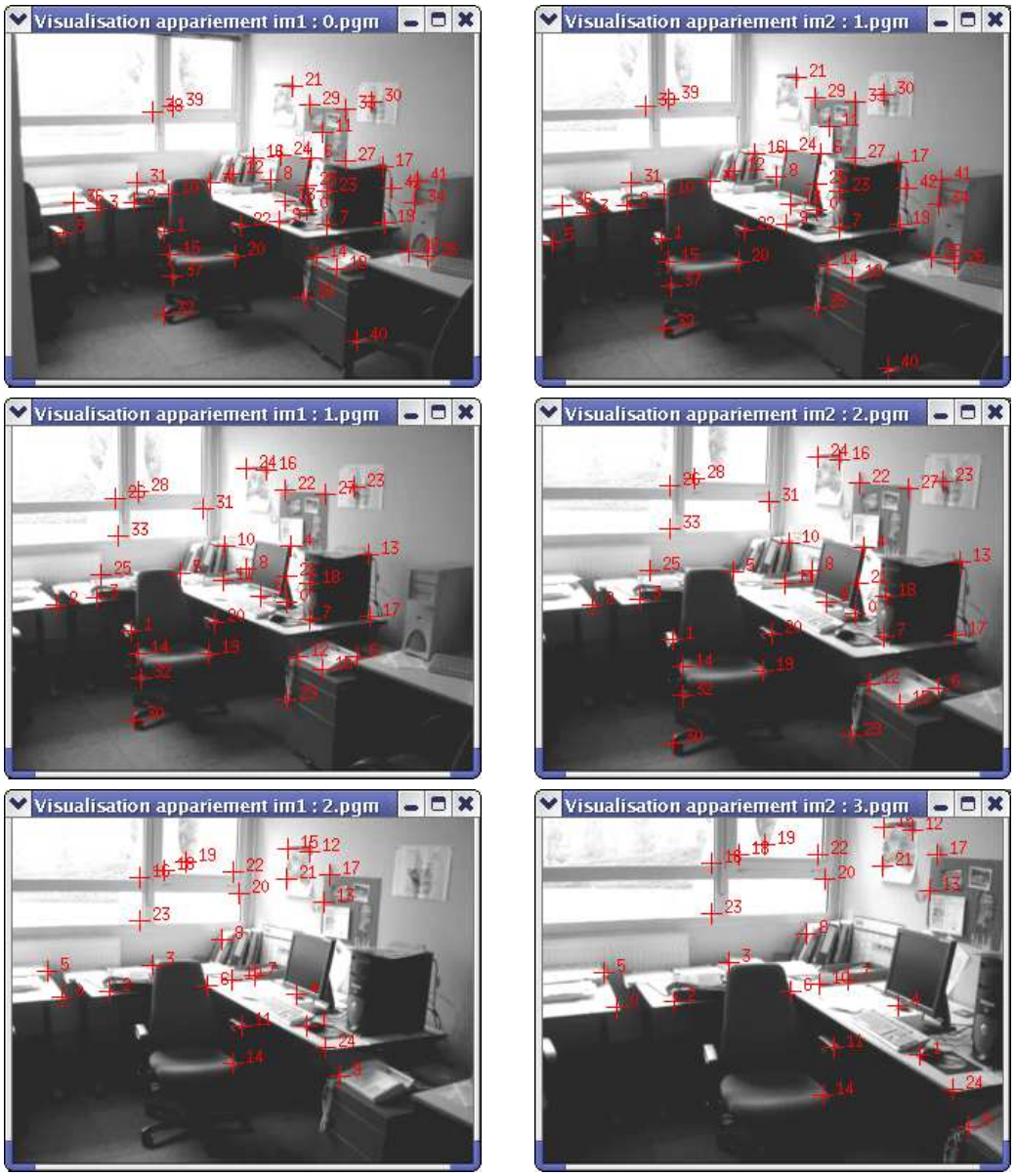
Durant le déplacement, le suivi d'un point peut échouer (le mouvement par rapport à l'image précédente est trop important, le signal au voisinage du point dans la nouvelle image n'est pas assez discriminant, ...). La méthode que nous proposons permet dans ces cas de réeffectuer une tentative de relocalisation de ce point dans les prochaines images fournies par la caméra (c'est le cas par exemple du point de l'ensemble $\mathcal{M}_0$, désigné par la flèche verte 1 dans l'image ψ_{100}).

Le contrôle de la validité des points localisés par reprojection ne permet pas d'éviter certaines erreurs. Par exemple, le point de l'ensemble $\mathcal{M}_6$ dans l'image ψ_{300} (flèche numéro 3) ne correspond pas à un point apparié entre les vues ψ_5 et ψ_6. La phase de contrôle par estimation du modèle affine n'a pas permis de rejeter ce point. Dans ce cas, rien ne permet dans l'état actuel de détecter que ce point n'est pas correct.

Pour minimiser ce problème, tous les points suivis (donc considérés comme visibles et associés à des primitives du chemin d'images) sont contrôlés périodiquement en estimant le modèle affine. Ce contrôle permet de détecter au cours du temps qu'un point initialement considéré comme correct ne l'est plus. Dans le cas du point cité ci-dessus, nous voyons que dans la dernière image ψ_{327}, celui-ci correspond bien à une des primitives de $\mathcal{M}_6$. Le contrôle de la validité de ce point entre les vues ψ_{300} et ψ_{327} a permis au système d'invalider le point dans un premier temps, et de retrouver la bonne position du point par reprojection dans un second temps.

Un autre exemple est celui du point de l'ensemble $\mathcal{M}_2$ observée dans vues ψ_{100} (flèche numéro 2), ψ_{150} et ψ_{200}. Dans l'image ψ_{100}, la position estimée par reprojection est incorrecte. Dans les vues ψ_{150} et ψ_{200}, la position suivie est valide. Le système a donc détecté entre les vues ψ_{120} et ψ_{150} qu'il suivait un mauvais point, et effectué ensuite une relocalisation de ce point.

La figure 3.12 montre l'évolution en pourcentage du nombre de points visibles pour chaque ensemble $\mathcal{M}_i$. Comme on pouvait s'y attendre, les premiers ensembles voient leur nombre de points visibles diminuer au cours du temps. Parallèlement, le nombre de primitives visibles des derniers ensembles augmente lorsque le robot se rapproche des zones où les prises de vue correspondantes du chemin ont été acquises.



$\mathcal{M}_0$

$\mathcal{M}_1$

$\mathcal{M}_2$

FIG. 3.9: Chemin d'images décrivant l'environnement (ensembles de $\mathcal{M}_0$ à $\mathcal{M}_2$)

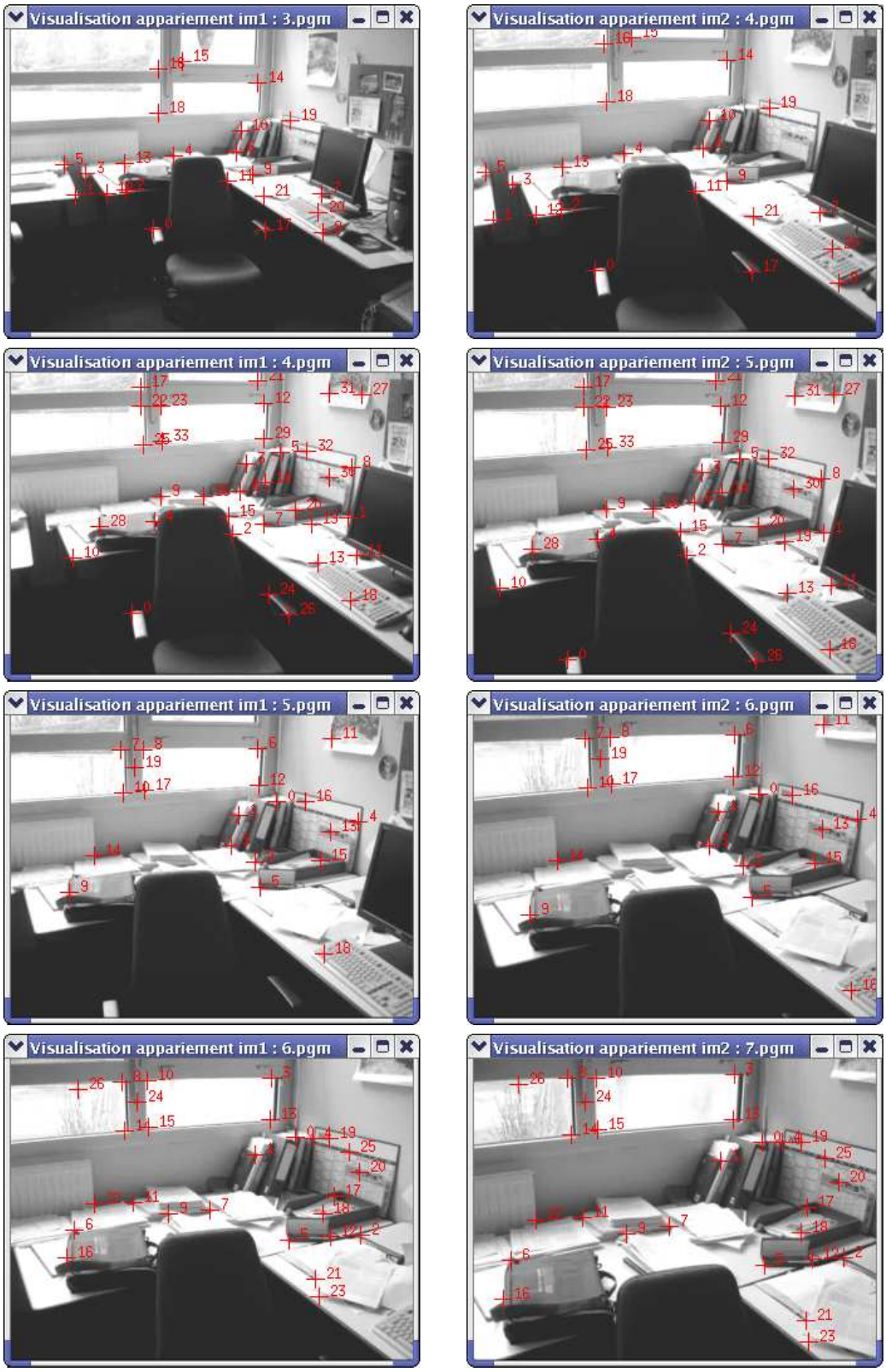


FIG. 3.10: Suite du chemin d'images (ensembles de $\mathcal{M}_3$ à $\mathcal{M}_6$)

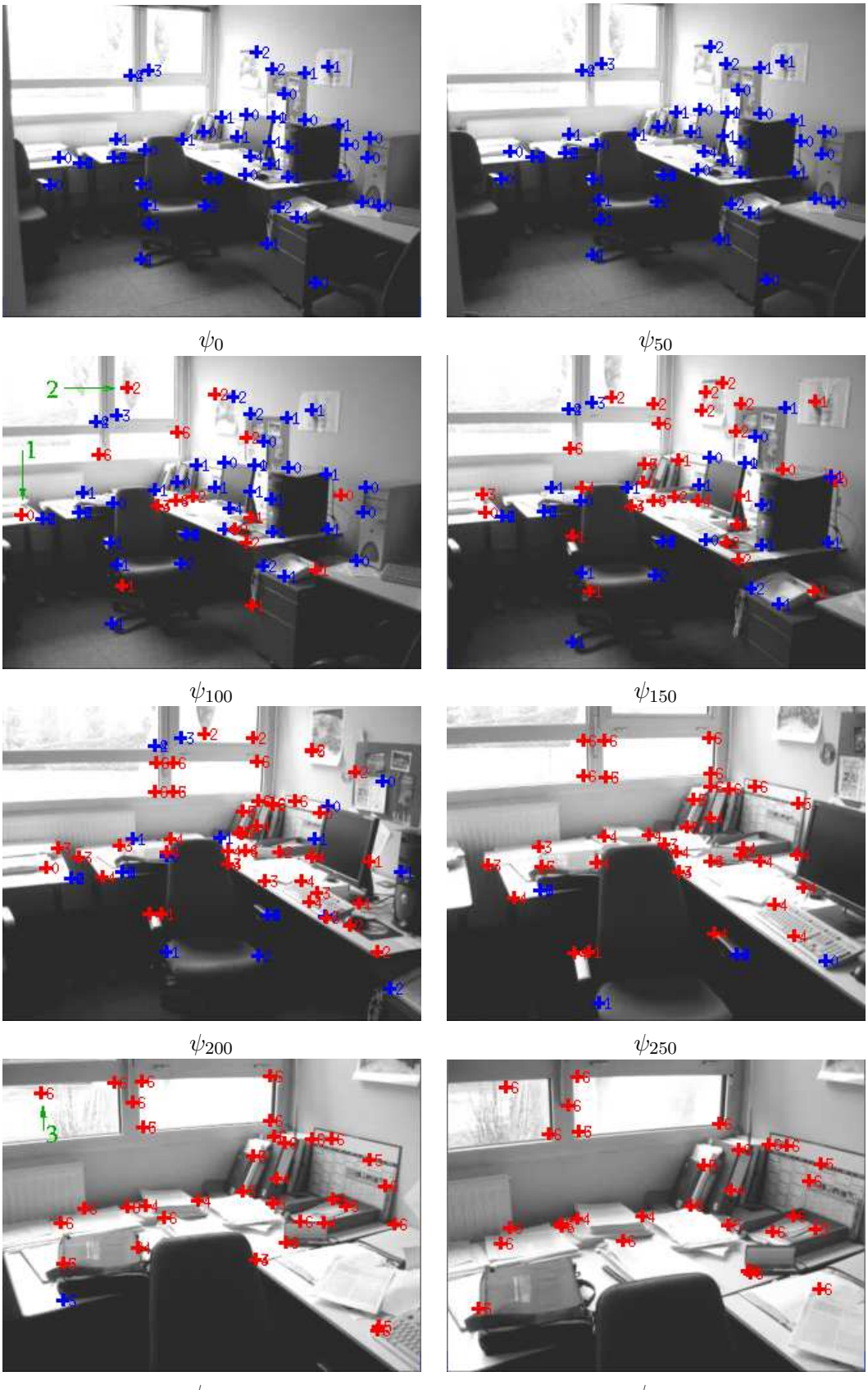


FIG. 3.11: Visualisation de prises de vue durant la séquence

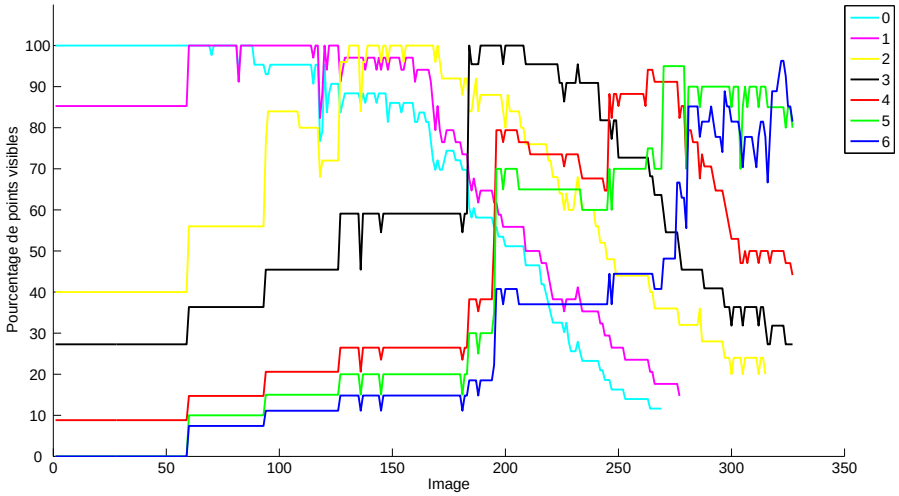


FIG. 3.12: Évolution du nombre de points visibles

Les figures 3.13 et 3.14 présentent un chemin d'images d'un environnement extérieur. Le déplacement réalisé correspond principalement à un mouvement de rotation de la caméra. Les figures 3.15 et 3.16 présentent quelques images ainsi que les points considérés comme visibles durant la navigation. La même signification que pour l'expérience précédente est associée au jeu de couleurs des points. Là encore, quasiment tous les points visibles dans la dernière prise de vue ont été localisés par reprojection. La présence de mauvaises détections n'a encore une fois pas empêché l'insertion de nouvelles primitives correctes.

Notons de plus que pour cette expérience, les paramètres intrinsèques utilisés ne correspondent pas à ceux de la caméra ayant acquise la séquence d'images. La technique de reprojection apparaît ainsi robuste aux erreurs de modélisation.

Le schéma 3.17 présente l'évolution des primitives visibles. On observe que certains points sont momentanément perdus, et relocalisés quelques images plus tard (c'est le cas pour l'ensemble $\mathcal{M}_2$ lorsque le nombre de points est proche de 100%, et de manière générale à chaque fois que l'on observe une diminution momentanée du pourcentage de points visibles.

Comme nous l'avons vu dans la section 3.2.1, une matrice d'homographie estimée entre deux prises de vue est relative à un plan de référence. Dans le protocole de mise à jour, une contrainte imposée est que ces trois points soient visibles dans l'image courante. Si ce n'est pas le cas, la matrice d'homographie n'est pas mise à jour.

Nous avons tenté lors d'expériences de changer de plan de référence, en choisissant trois point différents pour le définir. Les résultats n'étaient cependant pas assez satisfaisants pour permettre dans notre protocole un changement de plan de manière automatique.

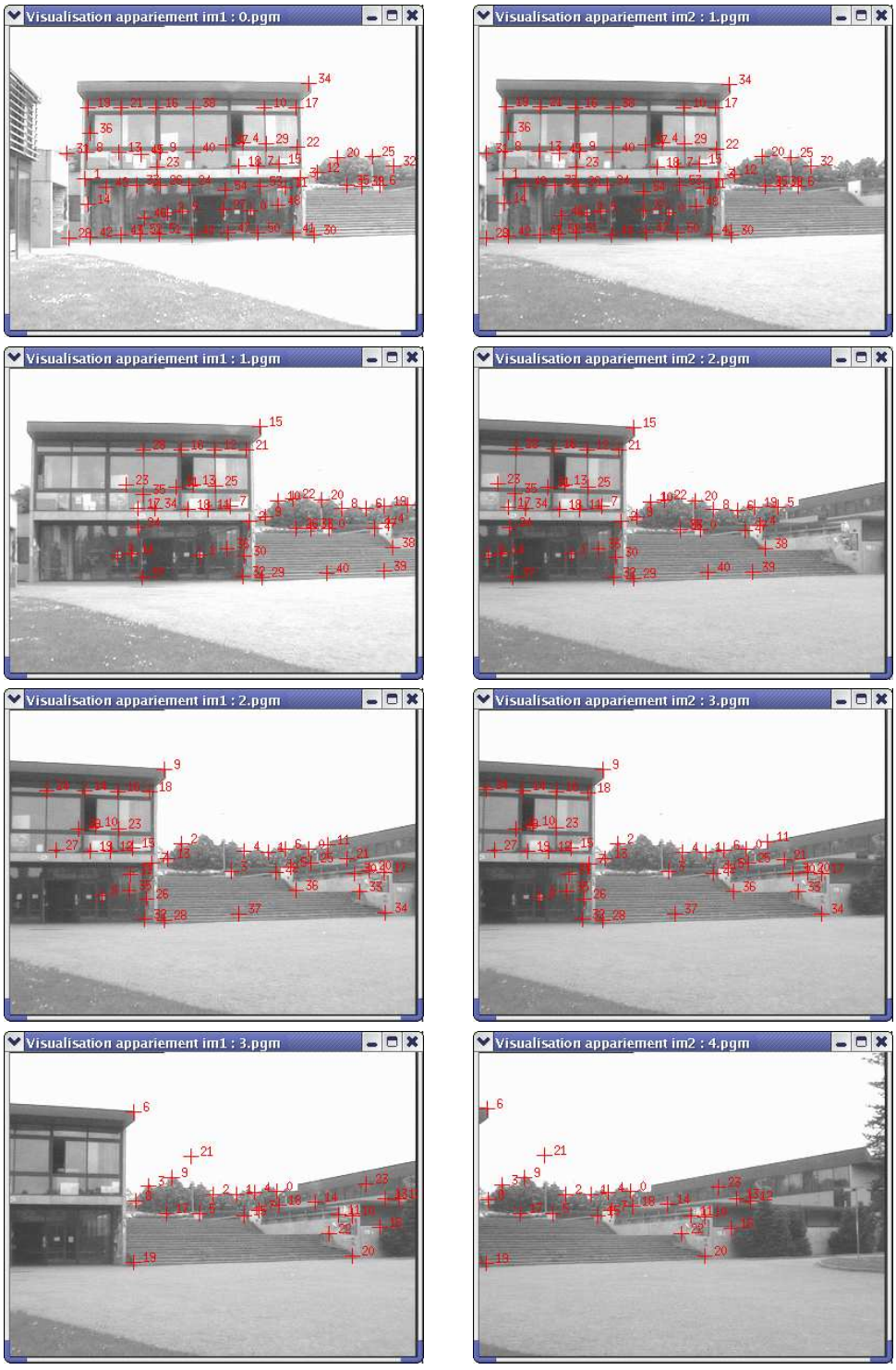
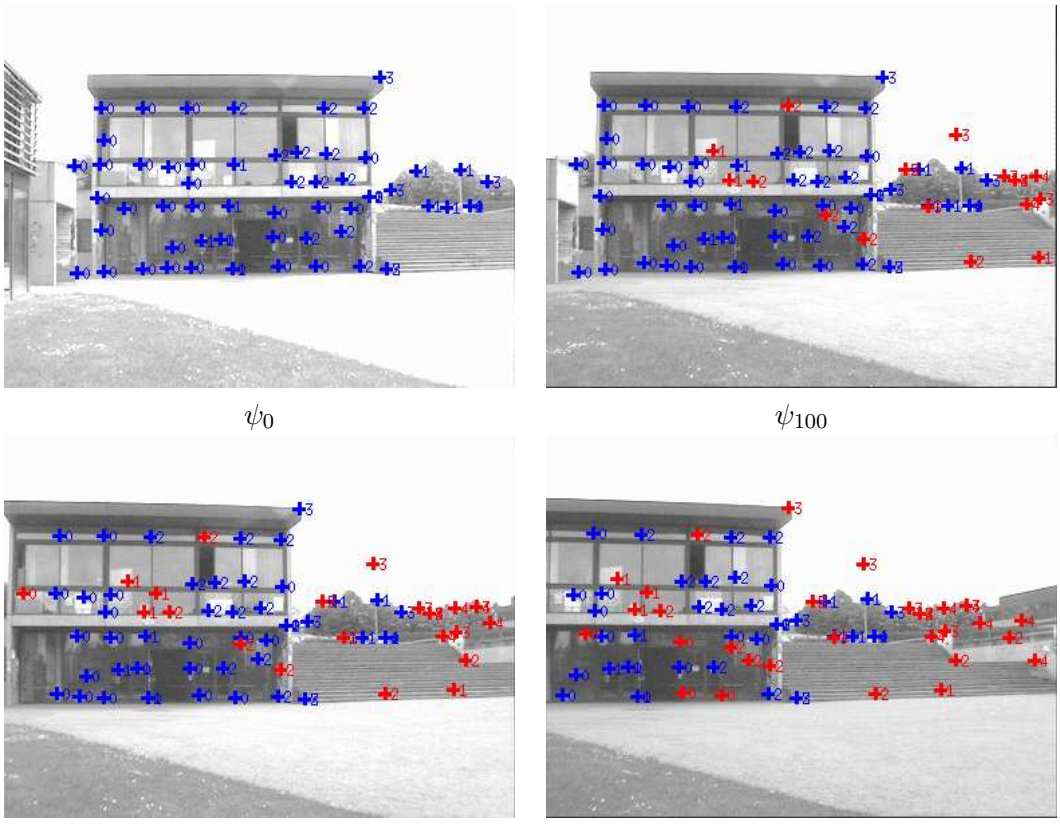


FIG. 3.13: Chemin d'images reliant les vues initiale (ψ_0) et désirée (ψ_5)

$\mathcal{M}_4$

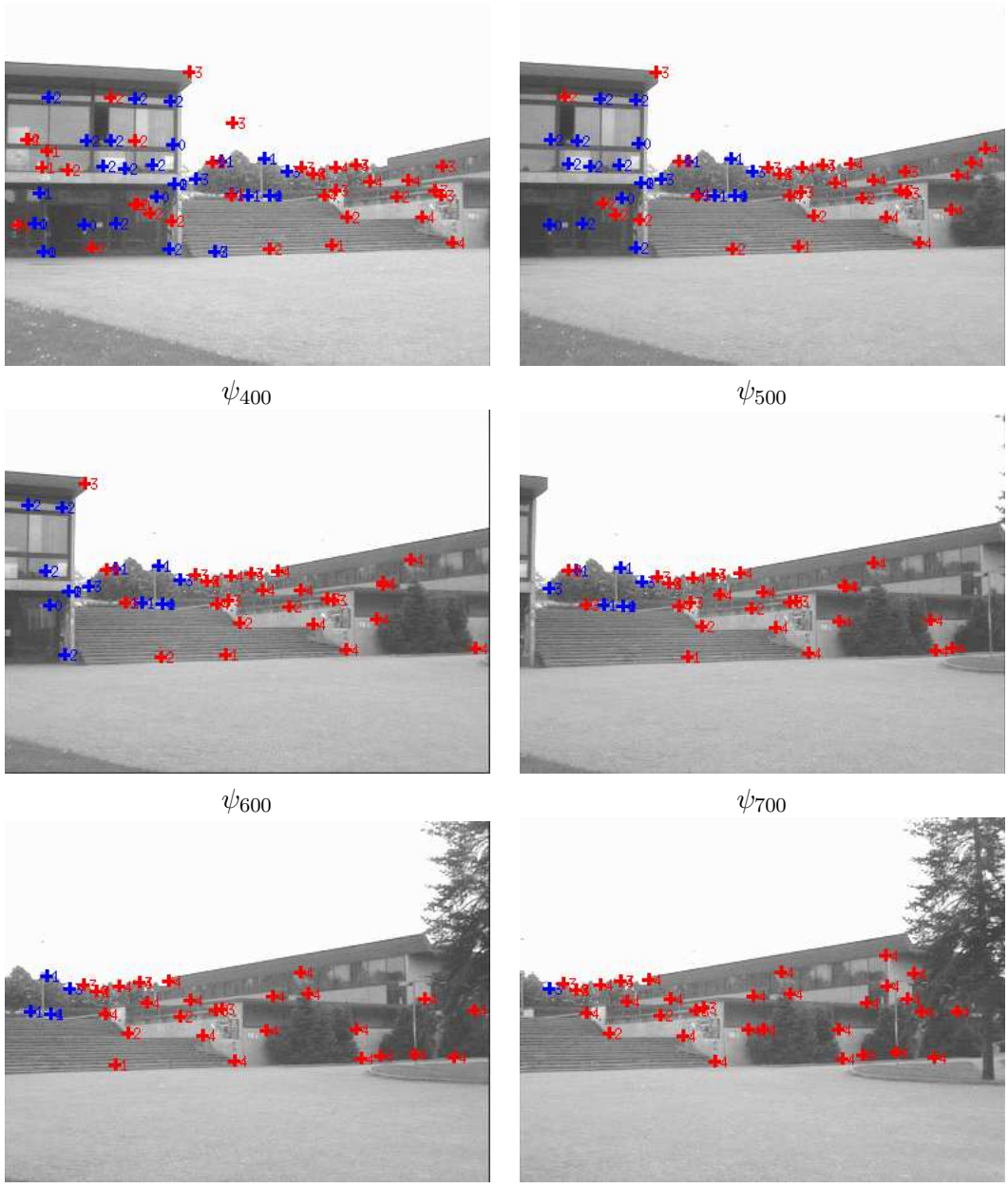
FIG. 3.14: Suite du chemin d'images reliant les vues initiale (ψ_0) et désirée (ψ_5)



ψ_0 ψ_{100}

ψ_{200} ψ_{300}

FIG. 3.15: Visualisation des prises de vue durant la séquence (vues ψ_0 à ψ_{300})



ψ_{400} ψ_{500}

ψ_{600} ψ_{700}

ψ_{800} ψ_{850}

FIG. 3.16: Visualisation des prises de vue durant la séquence (vues ψ_{400} à ψ_{850})

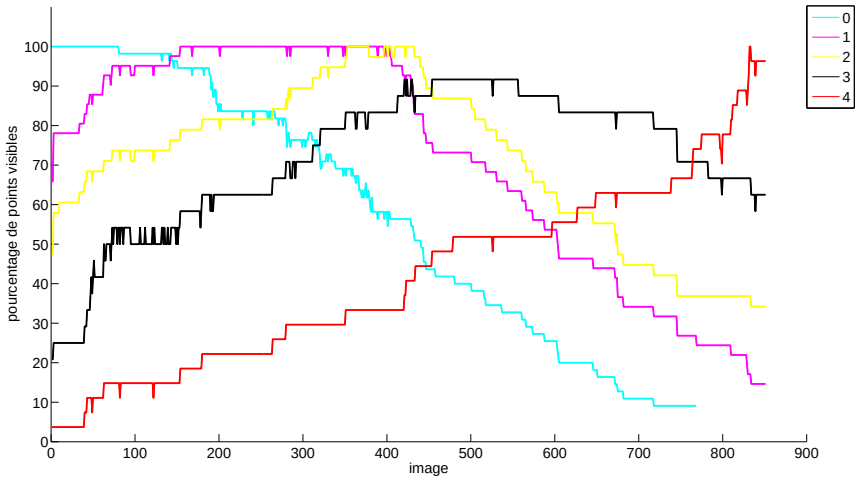


FIG. 3.17: Évolution du nombre de points visibles

3.3 Conclusion

Ce chapitre a présenté une méthode permettant de prendre en compte dynamiquement l'évolution de la scène observée par une caméra. Plus exactement, cette méthode gère la détection de l'apparition de nouveaux amers dans le champ de vision du capteur. Les amers utilisés correspondent à des points qui ont été appariés entre les différentes images du chemin.

La méthode proposée ne nécessite pas la connaissance du modèle $3D$ de la scène pour prédire l'apparition de nouveaux amers. Les relations traduisant la géométrie épipolaire sont suffisantes pour appréhender la structure $3D$ de la scène.

Certes, l'estimation de la matrice d'homographie et l'extraction du mouvement partiel associé revient quasiment à localiser la position du robot par rapport au chemin d'images (à un facteur d'échelle près pour la translation), et l'estimation des parallaxes à estimer une profondeur relative des primitives appariées entre deux images (ce qui correspond en soi à une reconstruction de la scène). Cependant, ces « reconstructions et localisations » implicites ne sont que locales puisque toujours associées à un couple d'images du chemin. Les résultats obtenus montrent en effet qu'il n'est pas impératif de se positionner par rapport à un repère global de la scène, et de connaître le modèle $3D$ de tout l'environnement pour réaliser la tâche de mise à jour des amers.

Dans ce chapitre, notre contribution a été de mettre en application, de manière dynamique et automatique, une méthode de transfert d'images qui a déjà fait ses preuves pour la création d'image virtuelle à partir de deux ou trois prises de vue d'une scène. L'absence d'information *a priori* sur l'environnement observé demande de nombreuses précautions et vérifications quant à la validité des reprojections effectuées. La méthode de recalage par estimation d'une transformation affine du signal dans le voisinage de la position estimée vers une fenêtre de référence permet d'effectuer ces contrôles.

Les résultats expérimentaux montrent cependant que ces contrôles n'empêchent pas totalement l'insertion de mauvais amers, c'est-à-dire la sélection d'un point de la vue courante qui n'est pas le projeté du point $3D$ initialement détecté dans le chemin d'images. Le contrôle de validité que nous effectuons reste local dans le sens où il ne traite qu'un point à la fois. La réalisation d'un contrôle plus global, concernant l'ensemble des primitives visibles, permettrait d'obtenir une mise à jour et un suivi plus robuste. L'utilisation de M-estimateurs [Huber 81] serait donc une piste à envisager pour améliorer cette méthode.

On trouve déjà dans la littérature des utilisations des M-estimateurs pour contrôler les résultats d'une phase de suivi de primitives. Dans [Plakas 00, Jin 01], une loi de rejet basée sur l'information photométrique associée aux différents points suivis permet de détecter les primitives incorrectes. Dans [Comport 03], dans le cadre de l'asservissement visuel de systèmes robotiques, une mesure géométrique correspondant à l'erreur entre les positions courante et désirée des primitives dans l'image est utilisée pour associer à chaque point un degré de confiance. Ce dernier permet ensuite de minimiser la contribution des points considérés comme incorrects dans le calcul de la loi de commande. Il serait donc intéressant d'utiliser les mêmes principes pour estimer la géométrie épipolaire entre les images du chemin et la prise de vue de la caméra, en relativisant la contribution de chacun des points en fonction de la confiance que l'on a dans celui-ci.

Il serait de plus intéressant de vérifier si l'utilisation d'informations *a priori* sur les mouvements réalisables par le système robotique peut entraîner une plus grande précision et robustesse dans l'estimation des différentes grandeurs en jeu (homographies, épipoles, parallaxes). Par exemple, un véhicule se déplace généralement sur un plan, et ne dispose que d'un seul axe de rotation. Ces contraintes sur le type de mouvement possible entre deux prises de vue pourraient donc être utilisées dès l'estimation de la matrice d'homographie.

Une contrainte posée par notre méthode réside dans la nécessité de toujours utiliser un même plan de référence de la scène pour l'estimation d'une matrice d'homographie entre deux prises de vue. Si théoriquement il est possible de changer le plan par rapport auquel est définie une matrice d'homographie, les résultats que nous avons obtenus lors d'expériences montrent que cette opération est relativement instable, et dans l'état actuel de notre méthode non envisageable de manière automatique. Cependant, nous pouvons espérer que les améliorations proposées ci-dessus permettront de rendre possible ce changement automatique de plan de référence.

De même, la prise en compte, comme information *a priori*, de la structure de la scène observée pourrait permettre de faciliter le choix et le changement du plan de référence. On peut en effet considérer que la plupart des environnements urbains respectent une « structure générique », où trois plans principaux se dégagent : celui du sol, et ceux des façades des bâtiments encadrant la route. Dans [Simond 04], ce genre d'information est exploité pour faciliter l'estimation des matrices d'homographies, et effectuer une localisation précise du véhicule. Dans notre cas, la prise en considération de ces informations lors du choix du plan de référence pourrait faciliter et rendre plus robuste le transfert d'images.

Notons enfin que notre méthode de mise à jour des amers visibles peut trouver d'autres applications que le cadre robotique dans lequel il est présenté dans ce manuscrit. Citons par exemple le cas de la réalité augmentée sans connaissance du modèle $3D$ de l'environnement, où la pose de la caméra est déduite de la géométrie épipolaire estimée entre la vue courante et une vue de référence [Pressigout 04, Simon 00]. Le formalisme que nous proposons permettrait de tolérer de

grands mouvements durant la séquence d'images ; le calcul en ligne des parallaxes des primitives suivies permettrait, comme une méthode de SLAM, de conserver un historique des amers détectés durant la séquence, et de les relocaliser automatiquement lorsque ceux-ci reviennent dans le champ de vision de la caméra. L'utilisation comme dans ce chapitre de plusieurs images de référence pourrait sans doute aussi favoriser la précision de la relocalisation d'amers qui étaient temporairement sortis du champ de vision de la caméra, mais aussi de rendre plus robuste l'estimation de la pose de la caméra, sans pour autant imposer que toutes les primitives appartiennent à un même plan.

Chapitre 4

Contrôle des mouvements d'un système robotique

Ce chapitre traite de la phase de commande d'un système de navigation autonome. Nous supposons ici qu'une phase préalable de localisation a été effectuée. La position initiale du robot ainsi que la position qu'il doit atteindre ont été mises en relation avec la représentation interne de l'environnement. Dans le cadre de travaux présentés dans ce manuscrit, la phase de localisation fournit un *chemin d'images* décrivant visuellement la partie de l'environnement que doit observer la caméra embarquée durant le déplacement vers la position désirée.

La première section présente les différentes approches de la navigation proposées dans la littérature. La deuxième section décrit la méthode que nous proposons, en accord avec le formalisme de localisation que nous avons établi dans le premier chapitre. La troisième section présente des résultats expérimentaux montrant le domaine de validité de notre approche. Une conclusion ainsi qu'une étude des différentes perspectives de ces travaux concluent ce chapitre.

4.1 La navigation robotique : état de l'art

Historiquement, la navigation robotique a tout d'abord été considérée comme un problème de planification de trajectoires. Le modèle $3D$ de la scène est supposé connu et des stratégies globales ou locales permettent de définir la séquence de configurations que doit prendre le robot pour atteindre son objectif.

Depuis le début des années 80, les avancées technologiques et algorithmiques permettent de contrôler directement les mouvements d'un système robotique à partir des informations perçues par ses capteurs. En particulier, les informations visuelles fournies par une caméra peuvent être utilisées pour définir une loi de commande du robot sans pour autant nécessiter le modèle de la scène. Les techniques dites d'asservissement visuel sont basées sur ces principes.

Plus récemment, les recherches en robotique mobile ont considéré des environnements de navigation très vastes, à tel point que les capteurs (caméra ou même laser) ne peuvent les décrire que localement à un instant donné. Une représentation interne de l'environnement de navigation devient alors nécessaire. Elle permet dans ce cas d'étendre le champ d'action des capteurs du système robotique, mais aussi de définir un ensemble de stratégies de navigation adaptées à l'environnement traversé.

Cette partie décrit plus en détail ces trois méthodes proposées pour le contrôle des mouvements d'un robot dans le cadre d'une tâche de navigation. Nous avons choisi de ne pas détailler les schémas de navigation utilisés dans les méthodes SLAM. En effet, dans ce type d'approche, le robot ne se déplace pas vers une position désirée, mais pour construire une description la plus complète possible de l'environnement. Le leitmotiv des déplacements ne permet donc pas de résoudre le problème que nous nous sommes posé durant cette thèse.

4.1.1 Planification de trajectoires

Les méthodes de planification de trajectoires que nous présentons ici correspondent aux techniques basées sur la connaissance du modèle de l'environnement de navigation. Nous avons déjà parlé des différentes méthodes de représentation de l'environnement dans la section 2.1.1.1.

Soit $\mathbf{p}(t)$ la pose de la caméra par rapport à un repère de la scène à l'instant t. La planification de trajectoires revient à déterminer une fonction continue sur l'horizon temporel :

$$\begin{cases} \Gamma & : & [0,1] & \rightarrow & \mathcal{C} \\ & & t & \mapsto & \mathbf{p}(t), \end{cases}$$

où $\mathbf{p}(0)$ désigne la position initiale du système robotique, et $\mathbf{p}(1)$ la position qu'il doit atteindre. Certaines méthodes de planification considèrent l'espace des configurations dans sa globalité pour déterminer Γ. D'autres n'utilisent qu'une représentation locale de l'environnement. Les stratégies employées dans les deux cas sont présentées ci-dessous.

4.1.1.1 Les méthodes globales

La principale difficulté des méthodes de planification globales réside dans la modélisation de l'environnement de navigation. Une fois le modèle obtenu, la planification d'une trajectoire est quasiment immédiate.

Nous rappelons que ces méthodes utilisent un graphe pour modéliser la scène. Ce graphe définit la connexité entre des courbes mono-dimensionnelles ou bien entre des cellules libres, suivant la représentation choisie.

Les courbes constituant une carte de l'environnement sont considérées comme un ensemble de trajectoires standard du système robotique. Un chemin optimal entre deux points de ces courbes est aisément obtenu par une recherche de chemin dans le graphe [Froidevaux 92]. La planification de trajectoires se résume donc à la connexion des configurations initiale et désirée avec le graphe, suivi de la recherche du plus court chemin entre ces deux positions.

Les représentations basées sur les cellules libres définissent initialement ces dernières de telle sorte que les déplacements inter-cellule et intra-cellule puissent être aisément obtenus. La planification revient donc à rechercher dans le graphe d'adjacence un chemin entre les deux cellules auxquelles appartiennent les configurations initiale et désirée.

Ces stratégies sont complètes : si une solution existe, elle sera trouvée ; si aucune solution n'existe, le système le spécifiera (à noter que la décomposition approximative en cellule peut, dans des cas extrêmes, ne pas trouver de solution même s'il en existe une).

Nous n'avons considéré ici que le problème de base de la planification de trajectoire. De nombreuses extensions ont été proposées pour prendre en compte la présence d'objets mobiles, pour rajouter au modèle les contraintes cinématiques et dynamiques du robot, pour donner la capacité au robot de déplacer ou saisir des objets, et pour modéliser l'incertitude. Les techniques correspondantes pourront être trouvées dans les ouvrages [Latombe 93, Laumond 01, Siegwart 04].

4.1.1.2 Les méthodes locales

Les méthodes locales ont la particularité de n'utiliser qu'une perception locale de l'environnement de navigation. À chaque pas de temps, les mouvements sont définis connaissant la position courante du système et la structure de l'environnement dans le voisinage proche du robot.

La technique des champs de potentiels est, sans doute, la méthode locale la plus utilisée. Elle a été initialement proposée par Khatib, pour réaliser de l'évitement d'obstacles en ligne [Khatib 86]. Dans ce formalisme, le robot est considéré comme une particule chargée soumise à un champ de potentiels. Un potentiel attractif attire le robot vers la configuration désirée, alors que des potentiels répulsifs l'éloignent des configurations interdites. Le robot se déplace dans le sens de la force induite, soit dans la direction opposée au gradient du potentiel. Ce type de méthode est particulièrement performant dans le cas où la tâche de navigation consiste simplement à atteindre une configuration désirée, sans plus de contraintes sur la trajectoire à suivre. Cependant, puisque le mouvement est défini à chaque instant et localement, il est difficile de prendre en compte des critères d'optimalité sur la globalité de la trajectoire à réaliser.

Comme toutes les méthodes locales (donc gloutonnes[1]), la méthode des potentiels souffre de l'éventuelle présence de minima locaux qui peuvent faire échouer la tâche de navigation. Dans [Koren 91], Koren et al ont montré que la présence d'obstacles dans la scène entraîne très facilement des oscillations du robot. Ces oscillations sont aussi présentes lors de la traversée de passages étroits. De plus, il n'existe pas dans le cas général de *fonctions globales de navigation*, possédant un seul point d´équilibre [Koditschek 87]. Rimon et Koditschek proposent de définir des fonctions de potentiel nommées fonctions de navigation. Celles-ci possèdent un seul minimum global et tous leurs minima locaux sont des positions instables du robot [Rimon 92]. Pour ce faire, les obstacles sont modélisés par des sphères auxquelles sont associées des potentiels répulsifs. La distance d'influence de ces potentiels dépend à la fois de la position des autres obstacles et de la position désirée. Certains travaux proposent de définir ces fonctions de navigation sur une décomposition régulière en grille de l'espace des configurations, ou encore sur un diagramme de Voronoï [Barraquand 92]. Cependant la connaissance du modèle de la scène devient alors impérative.

Certaines approches disposent de stratégies permettant de sortir le robot des puits de potentiel que sont les minima locaux. Les méthodes dites probabilistes [Barraquand 90, Bruce 02] effectuent une planification locale pour déplacer le robot et sauvegardent ses positions successives dans un graphe. Lorsque le robot atteint un minimum local, la position est rajoutée au graphe, et des mouvements aléatoires sont appliqués afin de sortir le robot de ce puits de potentiel [Overmars 95, Kavraki 96]. Ces méthodes sont connues pour être complètes de manière probabiliste : la probabilité de trouver un chemin vers la position désirée tend vers 1 lorsque le temps de

[1] en informatique le terme « glouton » caractérise un algorithme qui construit une solution de manière incrémentale, en choisissant à chaque pas de temps la décision qui semble localement la meilleure. Ce choix localement optimal n'a aucune raison de conduire à une solution globalement optimale, et peut même dans certains cas bloquer le processus dans des minima locaux.

calcul tend vers l'infini. Si ces techniques permettent de s'affranchir d'une prise en considération de l'espace de navigation dans sa globalité, elles imposent cependant encore la connaissance du modèle de la scène.

4.1.2 L'asservissement visuel

Contrairement aux techniques précédentes, l'asservissement visuel permet de contrôler en ligne les mouvements du robot. Cette méthode consiste à réaliser des tâches robotiques en intégrant dans la boucle de contrôle des informations visuelles fournies par une ou plusieurs caméras. La puissance des ordinateurs a permis en effet de diminuer considérablement les temps de traitement d'images. Il devient alors possible d'asservir le système robotique directement sur les informations extraites des capteurs vision embarqués ou déportés.

L'utilisation d'une boucle fermée permet de remettre à jour la commande appliquée au système robotique pour chaque image acquise et traitée. Un des grands avantages de l'emploi d'une telle boucle est que des erreurs de modélisation tant au niveau du système robotique que de la caméra n'entraînent pas nécessairement l'échec de la tâche robotique.

L'asservissement visuel peut être vu comme une approche basée vision de positionnement par rapport à un objet ou une scène. Dans le cadre de tâches de navigation, l'ampleur du mouvement à réaliser peut être très importante, provoquant une forte variation de l'information visuelle au cours du temps. Nous allons voir comment cette propriété est traitée en asservissement visuel. Mais avant de rentrer dans le vif du sujet, un rappel des notions de base de cette technique nous semble nécessaire. Nous renvoyons le lecteur intéressé vers [Chaumette 90, Hashimoto 93, Hutchinson 96, Chaumette 02] pour des descriptions plus détaillées.

4.1.2.1 Principe de l'asservissement visuel

Le but de l'asservissement visuel est d'atteindre et maintenir en temps réel une consigne visuelle. Nous présentons tout d'abord la relation liant la variation des informations visuelles perçues aux mouvements d'un système robotique. Nous verrons ensuite comment celle-ci est utilisée pour définir une loi de commande.

Modélisation des informations visuelles

L'asservissement visuel est basé sur la relation reliant la variation d'une mesure quelconque $\mathbf{s}(t)$ aux variations des paramètres de contrôle du système robotique. Tout ensemble de k informations visuelles peut être choisi si $\mathbf{s}(t)$ peut être définie par une application différentiable de SE_3 dans $\mathbb{R}^k$:

$$\mathbf{s} = \mathbf{s}(\mathbf{p}(t)), \tag{4.1}$$

où $\mathbf{p}(t)$ décrit la pose à l'instant t de la caméra par rapport à son environnement. Si l'on considère une scène fixe, et dans le cas d'une caméra embarquée, la différentielle de $\mathbf{s}$ dépend seulement de la variation de la pose $\mathbf{p}(t)$:

$$\dot{\mathbf{s}} = \frac{\partial \mathbf{s}}{\partial \mathbf{p}} \dot{\mathbf{p}} = \mathbf{L}_{\mathbf{s}} \mathbf{v}, \tag{4.2}$$

où :

- $\mathbf{L}_{\mathbf{s}}$ est appelée la *matrice d'interaction* associée à $\mathbf{s}$,
- $\mathbf{v}$ est le torseur cinématique de la caméra.

La structure de la matrice d'interaction dépend clairement des primitives visuelles choisies. Il est, par exemple, fortement conseillé (si ce n'est impératif) de choisir des primitives visuelles donnant une matrice d'interaction qui possède un bon conditionnement. Cette propriété, si l'on considère l'équation (4.2), permet d'assurer qu'un mouvement de la caméra va entraîner une variation de la mesure visuelle $\mathbf{s}$. Une forme intéressante, et donc recherchée, de la matrice d'interaction est celle d'une matrice diagonale et constante. Dans ce cas (idéal), le système est totalement découplé, c'est-à-dire qu'un mouvement sur un degré de liberté de la caméra entraîne la variation que d'une seule mesure visuelle.

L'espace de définition des primitives visuelles peut être l'espace image, l'espace de travail du robot, ou encore une combinaison des deux. L'espace image est le plus utilisé ; les primitives sont alors les coordonnées des points dans les images [Chaumette 90, Papanikolopoulos 73, Hager 97b], des droites [Chaumette 90, Andreff 00], ou encore des formes géométriques plus complexes comme des cylindres ou ellipses [Chaumette 90]. Lorsque la scène observée ne permet pas de détecter de telles primitives, les moments centraux de surface [Chaumette 04, Tahri 04], les contours représentés par des b-splines [Chesi 00] ne sont que des exemples des différentes mesures qui peuvent alors être considérées. L'utilisation d'informations de mouvement [SantosVictor 97, Crétual 98] permet de s'abstenir de l'extraction et du suivi de primitives particulières. Le choix d'un type de primitives dépend bien sûr des caractéristiques de la scène traitée.

Les informations visuelles exprimées dans l'espace de travail du robot sont déduites d'une reconstruction de l'environnement, de la localisation du système robotique, ou encore de l'estimation de son déplacement partiel. Dans [Martinet 96], les coordonnées de points $3D$ de l'objet d'intérêt sont utilisées. Généralement, la plupart des travaux considèrent la pose de la caméra comme information visuelle [Wilson 96, Daucher 97, Martinet 99].

D'autres travaux proposent de combiner dans le vecteur de mesure $\mathbf{s}$ des informations $3D$ et des informations $2D$. Là encore, la pose de la caméra peut être déduite de l'image courante si l'on connaît le modèle de l'objet [Corke 00a, Corke 00b], ou bien du mouvement partiel mesuré entre la vue courante et l'image désirée [Malis 99].

Loi de commande

Parmi les nombreuses lois de commande utilisées pour asservir un système robotique, nous nous limiterons ici à celles basées sur les fonctions de tâche, très souvent employées en asservissement visuel [Chaumette 90, Samson 91]. Les tâches robotiques sont considérées comme une régulation à zéro d'une fonction de tâche ou d'erreur, notée $\mathbf{e}$, et définie comme suit :

$$\mathbf{e}(\mathbf{p}(t)) = \mathbf{C}(\mathbf{s}(\mathbf{p}(t)) - \mathbf{s}^*), \tag{4.3}$$

où :

- la dimension de $\mathbf{e}$ est égale au nombre n de degrés de liberté à contrôler. La valeur de $\mathbf{e}$ dépend du temps et de la pose du capteur.
- $\mathbf{C}$ est une matrice de combinaison $n \times k$. Cette matrice permet d'utiliser un plus grande nombre d'informations visuelles que le système robotique n'a de degrés de liberté.
- $\mathbf{s}^*$ est la consigne, c'est-à-dire les valeurs que doivent atteindre les mesures visuelles courantes $\mathbf{s}$.

Si l'on connaît le modèle de l'objet considéré, la consigne $\mathbf{s}^*$ peut être déduite de la pose désirée par rapport à celui-ci. Si l'on ne possède cette information, $\mathbf{s}^*$ est obtenue lors d'une

phase d'apprentissage pendant laquelle le robot est amené à sa position désirée, et l'image correspondante mémorisée. Cette technique est la plupart du temps utilisée pour les asservissements visuels $2D$ (basés sur des informations dans l'espace image) et $2D1/2$ ($\mathbf{s}$ contenant des informations dans l'espace image et dans l'espace de travail). Un suivi de trajectoire planifiée ou bien d'objet en mouvement peut être obtenu en faisant évoluer la consigne $\mathbf{s}^*(\mathbf{t})$ au cours du temps.

Une décroissance exponentielle de l'erreur est assurée si :

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}, \tag{4.4}$$

où λ est un scalaire positif. La dérivée de $\mathbf{e}$ peut aussi être déduite de la relation (4.3). Si l'on considère que $\mathbf{C}$ est constante, nous obtenons :

$$\dot{\mathbf{e}} = \mathbf{C}\dot{\mathbf{s}} = \mathbf{C}\mathbf{L}_\mathbf{s}\mathbf{v} \tag{4.5}$$

Cette dernière équation associée à (4.4) permet de déduire les mouvements du robot à partir de l'erreur mesurée $\mathbf{e}$:

$$\mathbf{v} = -\lambda \left(\mathbf{C}\mathbf{L}_\mathbf{s}\right)^{-1} \mathbf{e} \tag{4.6}$$

Le choix des matrices $\mathbf{C}$ et $\mathbf{L}_\mathbf{s}$ dépend des informations visuelles utilisées. D'après les équations (4.4) et (4.5), la variation de $\mathbf{e}$ est régie par l´équation :

$$\dot{\mathbf{e}} = -\lambda \mathbf{C}\mathbf{L}_\mathbf{s}^T \left(\mathbf{C}\mathbf{L}_\mathbf{s}\right)^{-1} \mathbf{v},$$

ce qui impose de choisir les matrices de combinaison et d'interaction de manière à ce que la condition de positivité suivante soit autant que possible respectée :

$$\mathbf{C}\mathbf{L}_\mathbf{s} \left(\mathbf{C}\mathbf{L}_\mathbf{s}\right)^{-1} > 0$$

Dans le cas d'un asservissement visuel $2D$ avec redondance d'information, la matrice de combinaison peut être choisie comme $\mathbf{C} = \widehat{\mathbf{L}_\mathbf{s}^+}$ (où A^+ est la pseudo-inverse de la matrice A). La matrice d'interaction peut, quant à elle, être fixée comme étant la matrice d'interaction à la convergence $\mathbf{L}_{\mathbf{s}^*}$, ou bien remise à jour à partir des coordonnées courantes des informations visuelles $\mathbf{L}_{\mathbf{s}(t)}$. Une étude de convergence dans [Malis 04] a montré que les meilleurs résultats sont obtenus pour la moyenne de ces deux matrices.

4.1.2.2 L'asservissement visuel face aux grands déplacements et à la contrainte de visibilité

Comme nous l'avons vu, les mouvements du robots sont déduits de la différence entre la mesure visuelle courante et la mesure visuelle désirée. Il est donc nécessaire que la visibilité des primitives visuelles soit assurée durant tout le mouvement du système robotique. Si l'asservissement visuel $3D$ permet d'avoir une bonne trajectoire dans l'espace de travail de robot, elle ne permet pas, dans ses versions de base, de contraindre les primitives d'intérêt à rester visibles durant toute la tâche de positionnement. En asservissement visuel $2D$, la trajectoire de la caméra n'est pas contrôlée. Si le déplacement à réaliser est important, les primitives d'intérêt peuvent alors tout aussi bien sortir du champ de vision de la caméra.

Afin d'augmenter la zone de convergence de l'asservissement visuel, certains travaux proposent d'effectuer une planification préalable directement dans l'espace image. La trajectoire planifiée est ensuite réalisée en faisant évoluer la consigne $\mathbf{s}^*(t)$ au cours du temps. Notons, par

exemple, les travaux de Cowan et Koditschek [Cowan 99], qui utilisent, dans le cas d'un espace de travail plan et d'un espace image linéaire, une méthode globalement stable, en s'appuyant sur les principes des fonctions de navigation [Rimon 92]. Dans [Hosoda 95], une méthode de planification basée sur la géométrie épipolaire permettant d'éviter des obstacles est définie. Les trajectoires définies sont suivies par un robot doté d'un système stéréoscopique, pour atteindre une position désirée connue. Dans [Ruf 97], la planification est directement effectuée dans l'espace projectif. Les mouvements d'un système robotique sont observés par un système stéréoscopique faiblement étalonné. Différentes stratégies de mouvements sont présentées, et il est montré que les trajectoires réalisées sont optimales dans l'espace des configurations. Cependant, rien ne permet d'assurer que cette trajectoire n'entraîne pas la sortie de l'objet d'intérêt du champ de vision de la caméra. Dans [Mezouar 03], une décomposition sur l'horizon temporel de l'homographie $2D$ reliant la vue courante et la vue désirée est utilisée pour définir les trajectoires images des primitives. Le modèle de la scène n'est pas nécessaire, et les déplacements dans l'espace des configurations sont optimaux, mais la position de l'objet dans l'image n'est pas contrôlée.

D'autres approches prennent en compte la contrainte de visibilité durant la phase de planification. Dans [Mezouar 02a], une méthode de planification par champ de potentiels permet de contrôler la visibilité des points par l'ajout d'un potentiel répulsif adapté. Cependant, si ce schéma permet de contraindre les points à rester dans le champ de vision de la caméra, il ne peut gérer la présence éventuelle de points d'intérêt hors de ce cône de visibilité.

Notons que certains schémas d'asservissement visuel *en ligne* permettent aussi de considérer cette contrainte de visibilité. Ainsi, en asservissement visuel $2D1/2$, si les mouvements de translation selon $\vec{x}$ et $\vec{y}$ sont contrôlés par une primitive située au centre de gravité de l'objet d'intérêt, celle-ci a peu de chance de sortir du champ de vision, puisque son déplacement image correspond alors à une ligne droite de la position initiale à la position désirée [Malis 99, Martinet 99, Malis 02]. Cependant, aucune preuve analytique ne permet de l'assurer. Dans [Chesi 03], la visibilité durant un asservissement visuel $3D$ est contrôlée en étudiant successivement, avant de les effectuer, l'effet des mouvements de rotation et de translation sur la position des primitives. Si aucun de ces deux mouvements ne permet de garder les primitives dans le cadre de l'image, alors la caméra ouvre son champ de vision par un mouvement de recul le long de l'axe optique.

Finalement, si de nombreuses stratégies permettent de contraindre la caméra à garder les primitives d'intérêt dans son champ de vision durant la navigation, très peu de méthodes permettent une évolution de l'ensemble de points d'intérêt au cours du déplacement. Notons cependant les travaux de [Garcia 04], qui tolèrent la sortie du champ de vision de quelques points en intégrant dans la loi de commande une pondération sur les mesures d'erreur. Cependant, si ce formalisme permet de compenser la sortie de quelques primitives, il est impossible à appliquer pour des cas où de nombreuses primitives, voire même toutes, sont amenées à disparaître et d'autres à apparaître.

Si l'asservissement visuel est performant pour les tâches de positionnement ou de suivi, il nécessite un apport d'information pour réaliser des tâches de navigation. En effet, l'asservissement visuel ne peut être appliqué directement dès lors que les primitives constituant la mesure initiale $\mathbf{s}$ n'ont rien en commun avec les primitives définissant la consigne $\mathbf{s}^*$. C'est pourquoi une représentation interne de l'environnement est nécessaire afin de créer le lien entre les mesures visuelles initiale et désirée.

La partie suivante présente les techniques de navigation basées sur l'utilisation d'une ou plusieurs caméras et sur une représentation interne de l'espace de navigation. Comme nous le verrons, l'asservissement visuel n'est pas abandonné pour autant par les méthode décrites en 4.1.3 ; ses principes sont souvent utilisés pour réaliser des tâches de suivi spécifiques. Notons, de plus, que cet absence de recouvrement entre les primitives observées depuis les positions initiale et désirée n'handicape pas seulement les systèmes robotiques dotés d'un capteur de vision. Mis à part le GPS, aucun des capteurs extéroceptifs ne peut mettre en relation deux mesures associées à deux lieux totalement différents.

4.1.3 Une navigation basée vision

Des synthèses détaillées des schémas de navigation proposées dans la littérature sont réalisées dans [Franz 00, DeSouza 02]. Nous nous limiterons ici à ceux qui utilisent un capteur de vision. Deux groupes de méthodes se dégagent : la navigation associative, et la navigation par suivi.

Comme leur nom l'indique, les méthodes associatives « associent » une action à chaque position de l'environnement. Une fois localisé, le robot cherche dans sa mémoire le mouvement associé à sa position. On retrouve un tel schéma d'exécution dans [Matsumoto 00a, Matsumoto 00b]. L'image acquise par une caméra omnidirectionnelle est localisée dans la ou les séquences préalablement apprises. Le robot effectue alors le mouvement associé à l'image de la base la plus semblable. Dans ce contexte, les informations visuelles ne sont pas utilisées pour contrôler les mouvements du robot, mais seulement pour le localiser. De ce fait, l'information de mouvement peut être considérée comme un ordre donné au robot, puisqu'il ne dispose que d'un seul mouvement possible tant que la même image de la base est considérée comme la plus proche.

Plutôt que des ordres, la base peut proposer au robot différents comportements comme autant de stratégies de navigation qui peuvent être employées successivement. Ces schémas donnent plus de liberté au robot, et lui permettent surtout de pouvoir réagir s'il s'éloigne de la trajectoire idéale. Les caractéristiques locales de l'environnement guident le choix de la stratégie de navigation à employer. Dans [Gaspar 00, Vassallo 00], deux stratégies sont à la disposition du système robotique : la navigation topologique et le suivi de trajectoire. La navigation topologique est utilisée lorsqu'une grande distance est à parcourir, et que la position exacte du robot n'a pas besoin d'être contrôlée. Ce cas de figure est appliqué lors de déplacements le long d'un couloir par exemple, où les mouvements cherchent seulement à garder le robot au milieu de ce corridor qui est délimité par deux lignes. Le suivi de trajectoire basé vision est quant à lui utilisé pour des déplacements critiques, comme pour traverser une porte, ou encore pour tourner. L'asservissement visuel $2D$ est alors utilisé pour que les primitives d'intérêts suivent des trajectoires prédéfinies. Dans [Katsura 03], la définition des stratégies est effectuée automatiquement lors de la phase d'apprentissage, en étudiant les mesures de positions du système robotique obtenu par odométrie. Les mêmes schémas de navigation sont considérés (navigation topologique et suivi de trajectoire). Le suivi de trajectoire est par contre réalisé en utilisant l'odométrie et un système GPS.

La navigation par suivi consiste à définir une trajectoire que doit effectuer le robot durant son déplacement. Cette trajectoire est souvent définie par les trajectoires images que doivent suivre les primitives d'intérêt. En fonction du modèle du capteur et du système robotique utilisé, une matrice

d'interaction est définie et les techniques d'asservissement visuel peuvent alors être utilisées [Gaspar 00, Gracias 03]. Dans la deuxième méthode de [Gaspar 00], l'asservissement visuel $2D$ permet de contrôler la position d'un ensemble d'amers dans le cas d'un robot doté d'une caméra panoramique et qui se déplace sur un plan. Dans [Gracias 03], un schéma d'asservissement visuel permet de contrôler la position du point au centre de l'image afin qu'un robot sous-marin suive une trajectoire prédéfinie sur une mosaïque du fond marin.

Finalement, la navigation par suivi revient à changer la position désirée à chaque instant (ou quasiment) de la boucle d'asservissement. À moins d'utiliser des techniques de transfert d'images comme dans [Mezouar 02c], il est nécessaire de sauvegarder ces positions pour chaque instant. Le volume de données nécessaire devient très conséquent si l'environnement de navigation est vaste.

Pour diminuer la quantité d'information nécessaire, une solution est de ne conserver qu'un sous-échantillonage des séquences d'apprentissage. Les techniques d'asservissement visuel peuvent toujours s'appliquer, et chaque image de la séquence correspond alors à une cible intermédiaire vers laquelle le robot doit se diriger. Dans [Rasmussen 96], l'erreur entre la position courante des points dans l'image et celle désirée dans l'image intermédiaire permet de définir une fonction de tâche pour un asservissement $2D$. Dans [Argyros 01, Burschka 01], l'erreur d'angle entre les positions courante et désirée de points extraits d'images panoramiques est utilisée. [Jones 97] considère quant à lui l'information de corrélation (ZNCC) calculée sur l'ensemble de l'image pour se diriger vers les images intermédiaires. Dans [Mezouar 02c], les méthodes de planification de trajectoires images pour l'asservissement visuel sont étendues pour effectuer des tâches de navigation. Lors de la planification, chaque image intermédiaire est considérée comme une position que doit atteindre le système robotique. Dans [Blanc 04], le contrôle d'un robot non holonome est effectué à l'aide d'une caméra dirigée vers le plafond qui est enrichi par des amers artificiels. La navigation est encore réalisée en considérant les images intermédiaires comme des positions cibles successives.

D'un point de vue vision, ces positions intermédiaires permettent de délimiter des portions de l'espace où un ensemble de primitives est visible et donc utilisable par le robot pour contrôler son mouvement. Si une technique de reprojection est utilisée (voir le chapitre 3 précédent), l'asservissement vers une image intermédiaire courante peut être stoppée si assez d'information dans l'image courante peut être mis en relation avec celles de la prochaine image intermédiaire [Rasmussen 96, Burschka 01, Jones 97]. Cependant, le robot est tout de même amené explicitement à se diriger vers ces différentes positions intermédiaires.

4.1.4 Positionnement

Les techniques de navigation par planification ont montré leur efficacité pour les applications où le modèle de la scène est déjà connu. La définition de trajectoires directement dans l'espace des configurations permet de considérer des tâches complexes comme la saisie d'objets tout en s'assurant d'éviter les obstacles présents dans la scène.

Mais le modèle de la scène n'est pas toujours disponible comme une information *a priori*. Les techniques d'asservissement visuel démontrent, dans ce cas, qu'il est possible de s'affranchir d'une phase coûteuse de reconstruction. Cependant, ces méthodes ne permettent pas à elles seules de gérer des tâches de navigation où les positions initiale et désirée sont décrites par des informations visuelles totalement différentes.

Dans la plupart des techniques de navigation basées vision, le déplacement est décomposé en un ensemble de positions clés que doit successivement atteindre le système robotique. Un tel *modus operandi* implique cependant de fortes contraintes, soit sur la description interne de l'environnement, soit sur la trajectoire réalisée par le système robotique. On peut ainsi considérer que les positions intermédiaires constituent une trajectoire « optimale »du système robotique. La description interne de l'environnement doit donc être suffisamment riche et précise pour disposer des meilleures positions intermédiaires pour n'importe quelle tâche de navigation, ce qui est difficilement envisageable. Si ce n'est pas le cas, la convergence du système robotique vers ces différentes positions intermédiaires peut entraîner le système robotique à réaliser des mouvements qui lui sont inutiles pour atteindre la position désirée.

Si l'on désire contrôler par une caméra les mouvements d'un robot durant une tâche de navigation, il est clairement nécessaire d'utiliser des vues intermédiaires afin de mettre en relation les informations extraites de l'image initiale avec celles associées à l'image désirée. En asservissement visuel $2D$, l'erreur mesurée entre les positions courante et désirée de primitives permet de contrôler les mouvements du robot. Par analogie, l'évolution des primitives visibles dans les différentes vues intermédiaires peut permettre de définir les mouvements du robot. Les vues intermédiaires ne sont plus considérées comme des positions à atteindre, mais comme des descriptions visuelles de l'environnement à traverser. La seule contrainte imposée à ce chemin d'images est qu'il doit correspondre à une description visuelle *continue* de la scène (cette contrainte est présente dans la plupart des techniques de navigation basée vision trouvées dans la littérature). En d'autres termes, chaque couple d'images successives du chemin doit posséder des primitives en correspondance. La partie suivante présente comment ce principe peut être appliqué pour contrôler les mouvements d'un système robotique.

4.2 Une navigation basée sur l'évolution du champ de vision de la caméra

La figure 4.1 illustre le principe de navigation utilisé pour contrôler en ligne les mouvements du système robotique. Les cadres du haut constituent les prises de vue du chemin d'images décrivant l'environnement que doit observer le système robotique durant son déplacement. ψ_0 est la vue fournie par la caméra avant le déplacement, et ψ_3 la vue que l'on cherche à atteindre. Chaque couple d'images successives possède un ensemble de points en commun, soit l'ensemble $\mathcal{M}_i$ entre les vues ψ_i et ψ_{i+1} du chemin. Implicitement, ces ensembles peuvent être vus comme la description de zones que doit successivement observer la caméra du système robotique.

Les cadres du bas décrivent l'évolution des images acquises par la caméra durant la navigation. Avant le début du déplacement, la vue fournie par la caméra correspond à la vue ψ_0 du chemin d'images. Naturellement, toutes les primitives mises en correspondance avec l'image ψ_1 (en rouge) sont visibles. De même, nous pouvons supposer qu'un sous-ensemble des primitives de $\mathcal{M}_1$ se projette aussi dans le cadre image de l'image initiale. Ces points sont dessinés en bleu sur ψ_0. Notons que même si les prises de vue ψ_0 et ψ_2 ont des primitives en commun, il est possible qu'un appariement automatique d'images n'ait pu les détecter (ce qui explique la présence dans le chemin de la vue ψ_1). En effet, les méthodes de mise en correspondance nécessitent une zone de recouvrement entre les images suffisamment importante pour parvenir à

détecter des appariements. Si une faible quantité de points sont en commun, le bruit engendré par les autres points d'intérêts des deux prises de vue peut rendre impossible la détection automatique de ces correspondances.

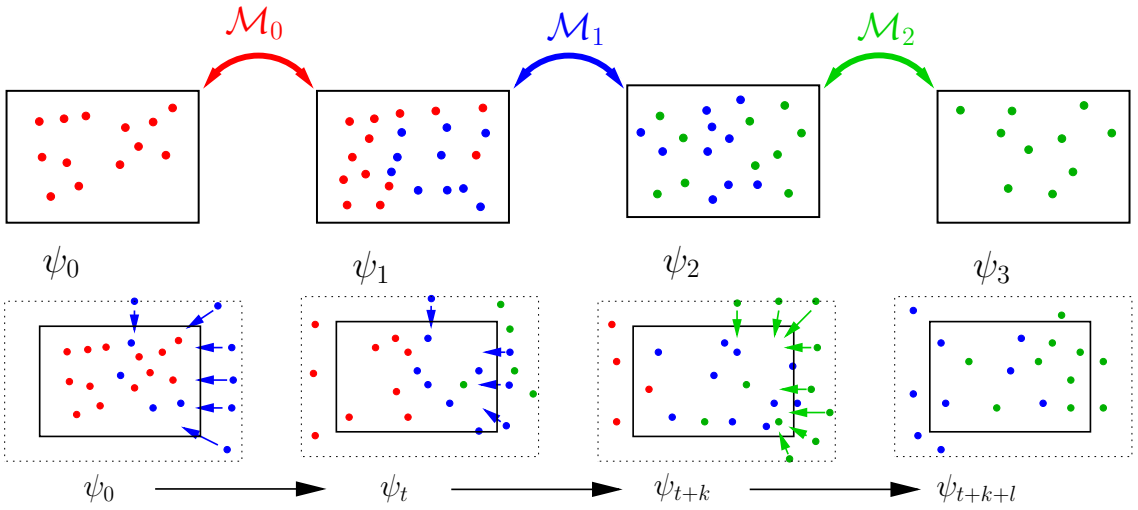


FIG. 4.1: Principe de la navigation basée sur l'évolution des primitives visibles

Le cadre d'une image n'est qu'une restriction matérielle du plan image infini de projection. Si l'on se dote d'outils de reprojection (comme ceux présentés dans le chapitre précédent), il est possible de déterminer la position sur le plan image courant de l'ensemble des primitives bleues mises en relation entre ψ_1 et ψ_2. Puisque ces primitives décrivent la prochaine zone que doit observer la caméra pour se rapprocher de sa position désirée, nous allons donc chercher par une loi de commande adaptée à faire rentrer ces primitives dans le champ de vision de la caméra.

Chemin faisant (vue ψ_t) les points rouges sortent peu à peu du champ de vision de la caméra. Ces primitives décrivent une zone que quitte le système robotique, il est donc normal que les points la décrivant disparaissent de l'image. De la même manière, les primitives décrivant la prochaine zone d'intérêt, soit $\mathcal{M}_2$, vont peu à peu rentrer dans le cadre de l'image (vue ψ_{t+k}). Le robot peut alors changer son ensemble d'intérêt, et chercher dorénavant à faire rentrer dans son champ de vision les primitives de $\mathcal{M}_2$. Lorsque suffisamment de primitives de cet ensemble sont visibles (vue ψ_{t+k+l}), nous pouvons considérer que le système robotique est proche de la position désirée associée à la vue ψ_3 du chemin. La tâche de navigation devient alors une tâche de positionnement ; elle peut être résolue par des techniques classiques d'asservissement visuel.

La méthode que nous proposons définit le déplacement que doit réaliser le système robotique pour chaque image acquise par la caméra. Notre loi de commande s'inspire des techniques de champ de potentiels. La section suivante rappelle les principes de ce formalisme. La section 4.2.2 définit les fonctions de potentiel utilisées dans la loi de commande (section 4.2.3). La section 4.2.4 présente le schéma de navigation complet utilisé pour diriger le robot vers une position désirée en fonction des informations fournies par la caméra, et de celles extraites du chemin d'images. Enfin des résultats viennent confirmer la validité de notre approche, tout en montrant ses limitations.

4.2.1 Formalisme du champ de potentiel

Bien que la méthode des champs de potentiel ait été initialement développée pour réaliser de l'évitement en ligne d'obstacle [Khatib 86], elle a trouvé ses plus nombreuses applications dans le cadre de la planification de trajectoire (comme nous l'avons vu dans la section 4.1.1.2).

Plusieurs stratégies de planification par champ de potentiel ont été proposées dans la littérature. Mais seule la technique par descente de gradient peut être utilisée en ligne si l'on ne connaît pas le modèle de la scène. Les propos que nous tenons ici concernent donc seulement cette méthode.

4.2.1.1 Généralités

Dans ce formalisme, le robot est représenté comme un point dans l'espace des configurations. Il est considéré comme une particule chargée sous l'influence d'un champ de potentiel V. La force induite par ce champ de potentiel définit le comportement du robot. Le potentiel V appliqué au système robotique dépend de la position $\mathbf{p}$ du robot dans l'espace des configurations $\mathcal{C}$:

$$\begin{cases} V & : & \mathcal{C} & \rightarrow & \mathbb{R} \\ & & \mathbf{p} & \mapsto & V(\mathbf{p}) \end{cases}$$

Ce potentiel est généralement défini comme la somme de deux composantes :

$$V(\mathbf{p}) = V_a(\mathbf{p}) + V_r(\mathbf{p})$$

V_a est un potentiel attractif, qui tend à amener le robot vers sa configuration désirée. Il atteint son minimum lorsque le robot atteint cette configuration. V_r est un potentiel répulsif qui permet de conserver le robot dans son espace libre $\mathcal{C}_{libre}$. Indépendant de la configuration désirée, ce potentiel prend en compte la structure locale de l'espace de navigation : sa valeur est nulle si le robot est éloigné des configurations indésirables, et augmente lorsqu'il s'en rapproche.

On peut noter que des minima locaux apparaissent lorsque ces deux potentiels sont opposés : $V_a(\mathbf{p}) = -V_r(\mathbf{p})$. Nous verrons par la suite que ce problème ne survient pas dans notre cas, puisque nous utilisons que des potentiels « attractifs » et quasiment découplés.

Un champ de potentiel induit une force $\overrightarrow{\mathbf{F}}$, définie comme la direction opposée du gradient du potentiel par rapport à la position de la particule :

$$\begin{cases} \overrightarrow{\mathbf{F}} & : & \mathcal{C} & \rightarrow & \mathcal{C} \\ & & \mathbf{p} & \mapsto & -\overrightarrow{\nabla}_{\mathbf{p}}^{\top} V(\mathbf{p}), \end{cases} \tag{4.7}$$

où $\overrightarrow{\nabla}_{\mathbf{p}} V$ représente le gradient de V par rapport à $\mathbf{p}$. Des deux potentiels V_a et V_r sont déduits deux forces $\overrightarrow{\mathbf{F}}_a(\mathbf{p})$ et $\overrightarrow{\mathbf{F}}_r(\mathbf{p})$ dont une somme judicieusement pondérée fournit la force appliquée au robot.

Dans le cadre de la planification par champ de potentiels, la trajectoire du robot est définie par un processus itératif. La pose courante $\mathbf{p}_k$ du robot permet de définir une force artificielle. Celle-ci est considérée localement comme la meilleure direction à suivre pour se diriger vers la position désirée tout en évitant les éventuels obstacles. La nouvelle position $\mathbf{p}_{k+1}$ du robot est obtenue en effectuant un déplacement d'amplitude ϵ dans la direction indiquée par la force $\overrightarrow{\mathbf{F}}(\mathbf{p}_k)$:

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \epsilon \frac{\overrightarrow{\mathbf{F}}(\mathbf{p}_k)}{\| \overrightarrow{\mathbf{F}}(\mathbf{p}_k) \|}$$

4.2.1.2 D'autres espaces de définition du potentiel

Le formalisme ci-dessus ne permet de prendre en compte que des potentiels définis dans l'espace des configurations du système robotique. Dans de nombreux cas de figure, il peut être plus aisé de définir un potentiel dans un espace transformé de $\mathcal{C}$. Dans [Mezouar 02b], des potentiels sont définis dans l'espace image, afin que la caméra garde dans son champ de vision un ensemble de points d'intérêt durant la navigation. De même, un autre potentiel est défini dans l'espace articulaire du robot afin d'éviter que celui-ci n'atteigne ses butées articulaires. Comme nous le verrons, certaines des fonctions de potentiel que nous utilisons sont définies dans l'espace image. Nous utiliserons donc le formalisme défini dans [Mezouar 02b] que nous rappelons ici.

Le déplacement du système robotique revient en terme mathématique à déterminer l'élément $\mathbf{p}$ permettant de minimiser la valeur de la fonction de potentiel. Une méthode numérique classique pour obtenir le minimum de V consiste à faire évoluer $\mathbf{p}(t)$ selon l'équation :

$$\dot{\mathbf{p}} = -\epsilon \mathbf{Q} \overrightarrow{\nabla_{\mathbf{p}}}^{\top} V, \tag{4.8}$$

où ϵ est un réel positif, et $\mathbf{Q}$ une matrice définie positive et considérée constante. En prémultipliant chaque membre par $\overrightarrow{\nabla_{\mathbf{p}}} V$, on obtient :

$$\frac{d}{dt} V(\mathbf{p}) = -\epsilon \overrightarrow{\nabla_{\mathbf{p}}} V \mathbf{Q} \overrightarrow{\nabla_{\mathbf{p}}}^{\top} V$$

V décroît au cours du temps tant que $\overrightarrow{\nabla_{\mathbf{p}}}^{\top}$ n'est pas nul, et est constant sinon. Si l'on choisit Q égale à la matrice identité, alors $\mathbf{p}$ évolue dans la direction opposée au gradient de V par rapport à $\mathbf{p}$. C'est le choix qui est effectué lorsque le potentiel est directement exprimé dans l'espace des configurations, comme dans la plupart des méthodes de planification par champ de potentiels.

Un potentiel peut être défini sur un espace transformé de $\mathcal{C}$ (comme par exemple directement sur le plan image). On note $V_{\mathbf{f}} = V \circ \mathbf{f}(\mathbf{p})$ une telle fonction de potentiel. $\mathbf{f}$ doit être une fonction dérivable sur tout l'espace des configurations. D'après la variation de $\mathbf{p}$ définie précédemment, les variations de $\mathbf{f}$ au cours du temps sont de la forme :

$$\dot{\mathbf{f}} = -\epsilon \left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \right) \mathbf{Q} \overrightarrow{\nabla_{\mathbf{p}}}^{\top} V = -\epsilon \left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \right) \mathbf{Q} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \right)^{\top} \overrightarrow{\nabla_{\mathbf{f}}}^{\top} V_{\mathbf{f}}$$

Pour que V évolue dans la direction opposée au gradient de V par rapport à $\mathbf{f}$, Q peut être choisie comme :

$$\mathbf{Q} = \mathbf{Q}_{\mathbf{f}} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \right)^{+} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \right)^{+T}$$

La matrice $\mathbf{Q}$ est ainsi bien définie positive, et on obtient pour la variation de $\mathbf{f}$ au cours du temps :

$$\dot{\mathbf{f}} = -\epsilon \overrightarrow{\nabla_{\mathbf{f}}}^{\top} V_{\mathbf{f}}$$

La force résultant d'un potentiel ainsi défini est alors :

$$\overrightarrow{\mathbf{F}_{\mathbf{f}}}(\mathbf{p}) = -\epsilon \mathbf{Q} \overrightarrow{\nabla_{\mathbf{p}}}^{\top} V = -\epsilon \left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}} \right)^{+} \overrightarrow{\nabla_{\mathbf{f}}}^{\top} V_{\mathbf{f}} \tag{4.9}$$

4.2.2 Potentiels attractifs basés sur l'évolution du champ de vision

Nous présentons ici les différents potentiels qui nous permettent de faire rentrer un ensemble de points d'intérêt dans le champ de vision de la caméra. Nous supposons ici connaître l'ensemble de points $\mathcal{M}_i$ que nous voulons observer avec la caméra. Nous verrons dans la section 4.2.4 comment cet ensemble peut être choisi.

Afin de simplifier les formules, nous représenterons par la suite les coordonnées d'un point dans l'image courante par $\mathbf{x}_j = [x_j \ y_j]$ au lieu de ${}^t\mathbf{x}_{p_j} = \left[{}^t x_j \ {}^t y_j \ 1\right]$.

Un point $\mathcal{X}_j$ se projette sur le plan image courant $\mathcal{I}_t$ en un point $\mathbf{x}_{pj} = [x_j \ y_j \ 1]$. Sa position sur $\mathcal{I}_t$ peut être obtenue de deux manières :
- par un algorithme de suivi de points si celui-ci était visible dans la vue ψ_{t-1} ;
- par transfert d'images sinon.

Dans le deuxième cas, l'homographie entre la vue ψ_t et ψ_i permet de déterminer la projection d'une primitive dans la vue courante :

$$\mathbf{x}_{p_j} \propto {}^t\mathbf{H}_{p_i} {}^i\mathbf{x}_{p_j} + \beta_j \mathbf{e}_i$$

Nous supposons, pour la définition des différents potentiels, que nous connaissons la position d'un ensemble de primitives $\mathbf{x}_j$ sur le plan image courant. Ces primitives sont visibles ou non dans l'image acquise par la caméra embarquée. Les potentiels que nous présentons visent à faire rentrer l'ensemble de ces points dans le champ de vision de la caméra.

4.2.2.1 Potentiel basé sur la visibilité

La projection $\mathbf{x}_j$ d'un point est considérée comme acceptable si $x_j \in [x_m + \alpha; x_M - \alpha]$ et $y_j \in [y_m + \alpha; y_M - \alpha]$, où x_m, x_M, y_m et y_M correspondent aux bords de l'image et α une constante positive permettant de définir une zone libre de projection $\mathcal{I}_{libre}$ à l'intérieur de l'image (voir la figure 4.2).

Pour un point, la fonction de potentiel doit être nulle sur tout l'espace libre $\mathcal{I}_{libre}$, et augmenter lorsque le point sort de la zone libre. Nous définissons un potentiel $V_{\mathbf{s}}(\mathbf{x})$ directement dans l'espace image de la forme :

$$V_{\mathbf{s}}(\mathbf{x}) = \sum_j V_{\mathbf{s}}(\mathbf{x}_j),$$

avec :

$$V_{\mathbf{s}}(\mathbf{x}_j) = g(x_j - x_M) + g(x_j - x_M) + g(y_m - y_j) + g(y_m - y_j), \tag{4.10}$$

et :

$$g(x) = \frac{1}{2}x^2 h(x) \ \text{ et } \ h(x) = \frac{\arctan(k\pi x)}{\pi} + \frac{1}{2} \tag{4.11}$$

k est un scalaire constant. La fonction $h(x)$ est la fonction arc-tangente normalisée sur $[0; 1]$. Elle correspond à la définition d'un pallier entre les valeurs 0 et 1. La position de transition correspond à $x = 0$. k permet de régler la courbure de la transition d'une valeur vers l'autre.

En choisissant comme positions de transition $(x_j - x_M)$, $(y_j - y_M)$, $(x_m - x_j)$ et $(y_m - y_j)$, ce potentiel est nul lorsque le point appartient à $\mathcal{I}_{libre}$ (voir la figure 4.3(a)). Il tend vers la fonction parabole lorsque le point s'éloigne de l'espace libre. Les quatre pics de potentiel observés sur la

figure correspondent à des positions où aucune des deux coordonnées de $\mathbf{x}_j$ n'est inclus dans son intervalle libre associé ($[x_m + \alpha; x_M - \alpha]$ ou $[y_m + \alpha; y_M - \alpha]$).

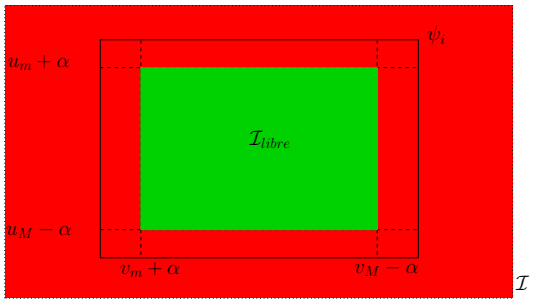
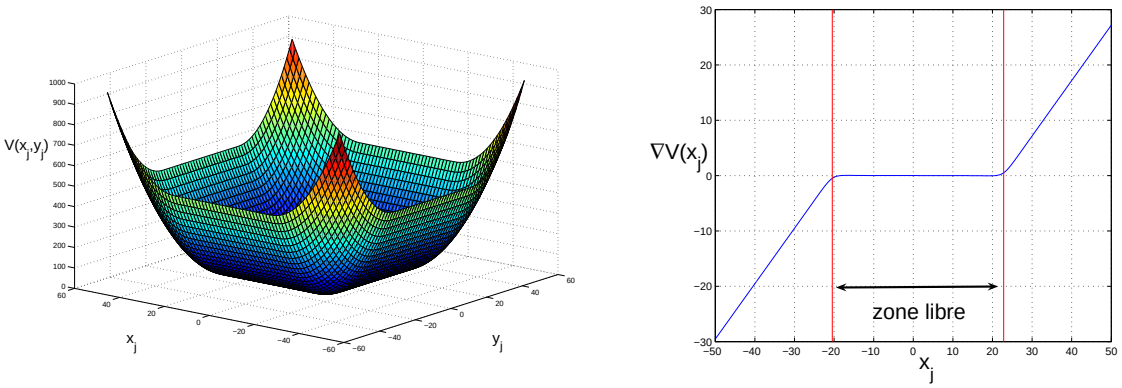


FIG. 4.2: L'espace libre $\mathcal{I}_{libre}$ est une restriction du plan image infini $\mathcal{I}$



(a) (b)

FIG. 4.3: Potentiel associé au champ de vision : (a) fonction de potentiel pour une primitive, (b) gradient du potentiel associé

La fonction de potentiel est continue et dérivable sur tout l'espace image. Son gradient est de la forme (dans le cas d'un seul point) :

$$\overrightarrow{\nabla_{\mathbf{s}}}^{\top} V_{\mathbf{s}}(\mathbf{x}_j) = \begin{bmatrix} (x_j - x_M)h(x_j - x_M) + (x_j - x_m)h(x_m - x_j) + \mathcal{O}(x_j - x_M) - \mathcal{O}(x_m - x_j) \\ \\ (y_j - y_M)h(y_j - y_M) + (y_j - y_m)h(y_m - y_j) + \mathcal{O}(y_j - y_M) - \mathcal{O}(y_m - y_j) \end{bmatrix}, \tag{4.12}$$

avec :

$$\mathcal{O}(x) = \frac{kx^2}{2(1 + k^2\pi^2x^2)} \tag{4.13}$$

Ce gradient est nul si le point appartient à $\mathcal{I}_{libre}$ et varie linéairement sinon (figure 4.3(b)). Si l'on considère cette seule mesure de potentiel, et si l'on travaille directement dans le repère courant de la caméra, la force associée est de la forme :

$$\overrightarrow{\mathbf{F}_{\mathbf{s}}}(\mathbf{p}) = -\lambda {\mathbf{L}_{\mathbf{s}}}^{+} \overrightarrow{\nabla_{\mathbf{s}}}^{\top} V_{\mathbf{s}} \tag{4.14}$$

$\overrightarrow{\nabla_{\mathbf{s}}}^{\top} V$ regroupe les gradients de potentiel de l'ensemble des points de l'ensemble $\mathcal{M}_i$ considéré :

$$\overrightarrow{\nabla_{\mathbf{s}}}^{\top} V_{\mathbf{s}} = \left(\overrightarrow{\nabla_{\mathbf{s}}}^{\top} V_{\mathbf{s}}(\mathbf{x}_1), \quad \ldots \quad , \overrightarrow{\nabla_{\mathbf{s}}}^{\top} V_{\mathbf{s}}(\mathbf{x}_n) \right),$$

avec $n = \text{card}(\mathcal{M}_j)$. $\mathbf{L_s}$ est la matrice d'interaction associée aux points de $\mathcal{M}_i$. Elle est de la forme :

$$\mathbf{L_s} = \mathbf{L}(\mathbf{s}, \mathbf{Z}) = \left(\mathbf{L}\left({}^t\mathbf{x}_1, Z_1\right), \quad \ldots \quad , \mathbf{L}\left({}^t\mathbf{x}_n, Z_k\right) \right), \tag{4.15}$$

où $\left({}^t\mathbf{x}_i, Z_i\right)$ est la matrice d'interaction associée au point $\mathbf{x}_i$. $\mathbf{L_s}$ peut encore être formulé de la manière suivante [Mezouar 01] :

$$\mathbf{L_s} = \mathbf{L}\left(\mathbf{p}, d_{i+1}\right) = \frac{1}{d_{i+1}} \begin{bmatrix} \mathbf{S} & \mathbf{Q} \end{bmatrix},$$

où $\mathbf{S} = (\mathbf{S}_1, \ldots, \mathbf{S}_k)$ et $\mathbf{Q} = (\mathbf{Q}_1, \ldots, \mathbf{Q}_k)$ sont deux matrices $2n \times 3$ indépendantes de d_{i+1}. Les matrices $\mathbf{S}_j$ et $\mathbf{Q}_j$ sont de la forme :

$$\mathbf{S}_j = \begin{bmatrix} -\frac{1}{\gamma_j} & 0 & \frac{x_j}{\gamma_j} \\ 0 & -\frac{1}{\gamma_j} & \frac{y_j}{\gamma_j} \end{bmatrix} \quad \mathbf{Q}_j = \begin{bmatrix} x_j y_j & -(1+x_j^2) & y_j \\ 1+y_j^2 & -x_j y_j & -x_j \end{bmatrix}$$

Le terme γ_j est déduit des équations (1.16),(1.17) et (1.18) :

$$\gamma_j = \frac{d_{i+1}}{Z_j} = \frac{r_j}{\rho_j} \tau_j.$$

À partir du mouvement partiel extrait de la matrice d'homographie ${}^{i+1}\mathbf{H}_{n_t}$ entre la vue courante ψ_t et la vue ψ_{i+1} du chemin d'images, le ratio γ_j d'un point de coordonnées normalisées ${}^t\mathbf{x}_{n_j}$ dans l'image courante ψ_t peut être défini comme étant (d'après l'équation précédente et les relations (1.16),(1.17) et (1.18)) :

$$\begin{cases} \gamma_j = {}^t\mathbf{n}^{\top} {}^t\mathbf{x}_{n_j} \left(1 + {}^t\mathbf{x}_{n_j}{}^{i+1}\mathbf{R}_t {}^{i+1}\mathbf{t}_t / d_t\right) & \text{si } \mathcal{X}_j \in \pi \\ \gamma_j = \frac{\|{}^{i+1}\mathbf{t}_t / Z_j\|}{\|{}^{i+1}\mathbf{t}_t / d_t\|} \left(1 + {}^t\mathbf{x}_{n_j}{}^{i+1}\mathbf{R}_t {}^{i+1}\mathbf{t}_t / d_t\right) & \text{si } \mathcal{X}_j \notin \pi \end{cases}$$

Un tel potentiel ne permet cependant pas de contraindre suffisamment les mouvements de la caméra. La figure 4.4 illustre ce fait. Plusieurs mouvements peuvent être réalisés afin de rendre visibles les points noirs (qui sont actuellement hors du champ de vision de la caméra). En particulier, toute combinaison des trois mouvements présentés (rotation autour de $\vec{x}$, translation suivant $-\vec{y}$ et $-\vec{z}$) permettra de rapprocher le cadre de l'image de ces points. Pour lever ces ambiguïtés, il est donc nécessaire de rajouter des contraintes sur les mouvements du robot permettant, suivant les cas de figure, de choisir le mouvement le plus adapté.

4.2.2.2 Potentiel basé profondeur

Le potentiel précédent permet de contraindre la caméra à se déplacer afin que toutes les primitives considérées se projettent dans la zone libre $\mathcal{I}_{libre}$. Cette zone libre correspond à l'intersection du cône de visibilité $3D$ avec le plan image $\mathcal{I}$, comme l'illustre la figure 4.5(a). Un point $3D$ de coordonnées $\mathbf{X} = (X, Y, Z)$ est inclus dans ce cône si (pour une focale égale à un mètre) :

$$X \in [Zu_m, Zu_m] \text{ et } Y \in [Zv_m, Zv_M]$$

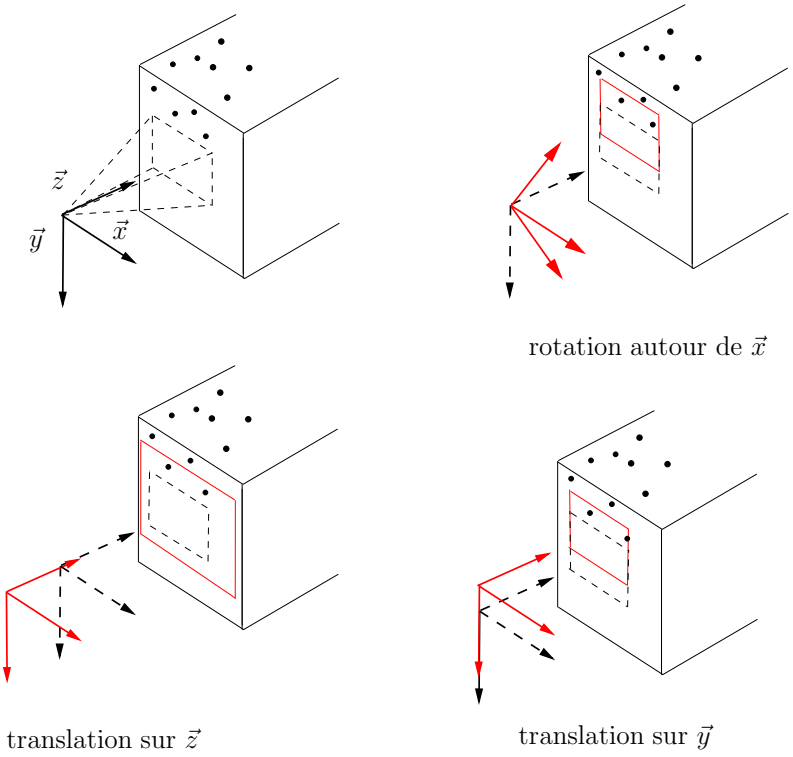


FIG. 4.4: Illustration des différents mouvements permettant de rendre visibles les points noirs qui sont hors du champ de vision de la première caméra (le cadre rouge correspond au cadre image obtenu suite au mouvement).

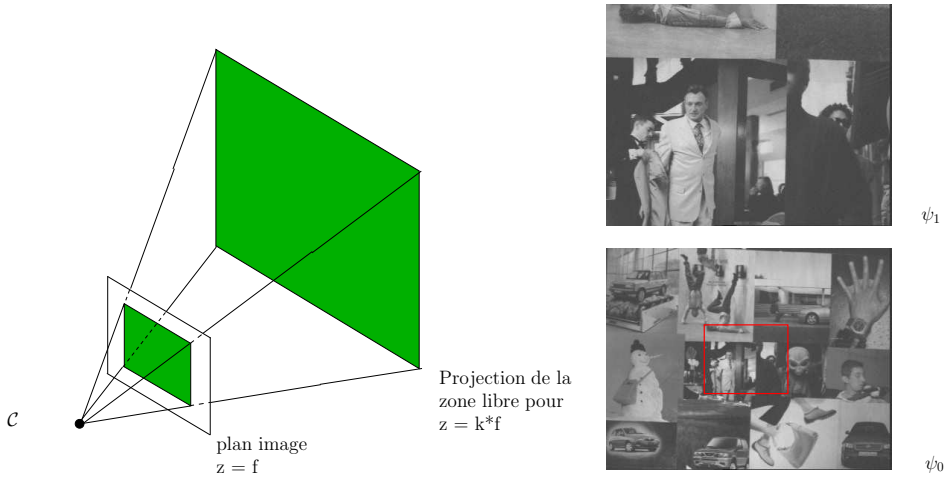


FIG. 4.5: Problème des déplacements le long de l'axe optique : (a) cône de visibilité engendré par la zone libre $\mathcal{I}_{libre}$, (b) comparaison des cadres images suite à un mouvement le long de l'axe optique

Sur la figure 4.5(a), si l'on considère que la distance Z correspond à la profondeur d'un point de la scène, la base du cône de visibilité correspond à la zone à laquelle doit appartenir le point pour être visible. Clairement, plus Z est grand, plus cette zone l'est aussi.

La figure 4.5(b) illustre un cas de déplacement le long de l'axe optique. La zone de visibilité de ψ_1 correspond à un espace de la scène beaucoup plus restreint que celle de ψ_0, puisque la distance de la scène à la vue ψ_1 est plus faible que celle à l'image ψ_0. Sur l'image ψ_0 est tracé en rouge le cadre de l'image ψ_1. Toutes les primitives potentiellement en correspondance entre ψ_0 et ψ_1 se retrouvent dans cette zone. Plus on recule la caméra, plus la projection de ce cadre diminue, et plus les primitives se regroupent au centre de l'image. Il est donc impossible de déplacer le robot avec le potentiel défini précédemment de ψ_0 vers ψ_1 puisque tous les points appartiennent déjà à la zone libre de ψ_0.

Visuellement, le déplacement de ψ_0 vers ψ_1 revient à faire augmenter la surface du cadre rouge jusqu'à ce qu'il corresponde à la surface d'une prise de vue. Nous proposons donc d'utiliser une telle information pour contrôler les mouvements le long de l'axe optique.

Plus exactement, nous nous basons sur les travaux présentés dans [Chaumette 04, Tahri 04], où les informations visuelles utilisées sont des moments calculés sur des objets plans. Des combinaisons judicieuses de différents moments ont permis d'obtenir des lois de commande quasiment découplées. En particulier, il a été proposé dans [Tahri 04] une mesure qui ne dépend quasiment que des mouvements le long de l'axe optique. Cette mesure est la suivante :

$$a_n = Z^* \sqrt{\frac{a^*}{a}}, \tag{4.16}$$

avec Z^* la profondeur désirée au plan de référence, a^* la valeur désirée de la grandeur a, composée des deux moments centrés d'ordre 2 :

$$a = \mu_{02} + \mu_{20},$$

avec, pour un ensemble de n primitives exprimées en coordonnées normalisées :

$$\mu_{ij} = \sum_{k=0}^{n} (x_k - x_g)^i (y_k - y_g)^j ,$$

où (x_g, y_g) correspond au centre de gravité image de ces n points :

$$\begin{pmatrix} x_g \\ y_g \end{pmatrix} = \frac{1}{n} \sum_{k=0}^{n} \begin{pmatrix} x_k \\ y_k \end{pmatrix}$$

Quand le plan de l'objet est parallèle au plan image, la matrice d'interaction associée à cette information visuelle est donnée par :

$$\mathbf{L}_{a_n}^{\parallel} = [0\ 0\ -1\ -\epsilon_1\ \epsilon_2\ 0], \tag{4.17}$$

avec :

$$\epsilon_1 = y_g + (y_g \mu_{02} + x_g \mu_{11} + \mu_{21} + \mu_{03})/a$$
$$\epsilon_2 = x_g + (x_g \mu_{20} + y_g \mu_{11} + \mu_{12} + \mu_{30})/a$$

Les grandeurs ϵ_1 et ϵ_2 peuvent être considérées comme négligeables par rapport à 1. La matrice d'interaction ci-dessus n'est exacte que si la position désirée de la caméra est fronto-parallèle à

l'objet supposé plan. Il est clair dans ce cas que des mouvements de translation le long des axes $\vec{x}$ et $\vec{y}$, ou bien encore de rotation autour de l'axe optique n'influent pas sur la valeur de a_n.

Bien que nous ne puissions pas supposer satisfaire les conditions énoncées ci-dessus, nous proposons d'utiliser cette mesure pour contrôler les mouvements le long de l'axe optique. Nous effectuons l'hypothèse que la considération d'un environnement tridimensionnel permet tout de même de garder un même ordre relatif de cette mesure. En effet nous ne cherchons pas avec cette mesure à atteindre une position exacte le long de l'axe $\vec{z}$.

Un même ensemble de point est utilisé pour le calcul de a_n et a_n^*. La mesure désirée a_n^* est effectuée dans l'image ψ_{i+1} du chemin, et la mesure a_n dans l'image courante. Puisque nous ne cherchons pas à atteindre exactement les différentes positions intermédiaires, nous ne voulons donc pas obtenir précisément la même mesure dans la vue courante que celle effectuée dans ψ_{i+1}. Nous cherchons plutôt à contraindre la mesure a_n à atteindre une zone de confiance définie autour de a_n^*. Notons $p \in [0,1]$ le pourcentage de liberté autorisé autour de la mesure a_n^*. Nous voulons donc que la mesure a_n soit telle que :

$$a_m = a_n^*(1-p) < a_n < a_n^*(1+p) = a_M$$

Cette contrainte peut s'exprimer avec le potentiel suivant :

$$V_a(a_n) = g(a_n - a_M) + g(a_m - a_n), \tag{4.18}$$

où $g(x)$ a été donnée en (4.11), page 120. Le gradient d'un tel potentiel est alors :

$$\overrightarrow{\nabla_a}^{\top} V_a = (a_n - a_M)h(a_n - a_M) + (a_n - a_m)h(a_m - a_n) + \mathcal{O}(a_n - a_M) - \mathcal{O}(a_m - a_n), \tag{4.19}$$

où $\mathcal{O}(x)$ est donné en (4.13). Sachant que :

$$a_n^* = Z^* \sqrt{\frac{a^*}{a^*}} = Z^*,$$

les grandeurs $(a_n - a_M)$ et $(a_m - a_n)$ deviennent :

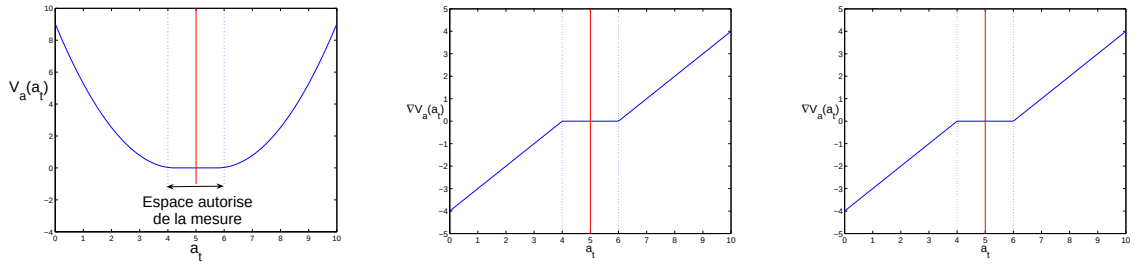
$$a_n - a_M = Z^* \left(\sqrt{\frac{a^*}{a}} - 1 - p \right) \quad , \quad a_m - a_n = Z^* \left(1 - p - \sqrt{\frac{a^*}{a}} \right)$$

En considérant que $k' = k/Z^*$, le gradient du potentiel peut encore s'écrire :

$$\begin{aligned} \overrightarrow{\nabla_a}^{\top} V_a = {} & Z^* \left(\sqrt{\frac{a^*}{a}} - 1 - p \right) h_{k'} \left(\sqrt{\frac{a^*}{a}} - 1 - p \right) + \mathcal{O}_{k'} \left(\sqrt{\frac{a^*}{a}} - 1 - p \right) + \\ & Z^* \left(\sqrt{\frac{a^*}{a}} - 1 + p \right) h_{k'} \left(1 - p - \sqrt{\frac{a^*}{a}} \right) - \mathcal{O}_{k'} \left(\sqrt{\frac{a^*}{a}} - 1 + p \right) \end{aligned} \tag{4.20}$$

avec $h_{k'}(x)$ et $\mathcal{O}_{k'}(x)$ correspondant aux fonctions initiales $h(x)$ et $\mathcal{O}(x)$ où k' remplace k. De manière pratique, les deux appels à la fonction $\mathcal{O}_{k'}$ sont quasiment égaux ; leur soustraction est ainsi quasiment nulle. Dans la fonction $h_{k'}(x)$, une mauvaise estimation de la profondeur influe simplement sur la pente de la transition. Une mauvaise estimation de Z^* n'influe donc que sur l'amplitude du gradient de potentiel $\overrightarrow{\nabla_a} V_a$.

La figure 4.6(a) présente ce potentiel. Il est nul lorsque la mesure est dans la zone libre, délimitée par les deux lignes verticales bleues autour de la valeur a_n^* présentée en rouge. Plus la mesure s'en éloigne, et plus le potentiel est fort. Les figures 4.6(b) et 4.6(c) présentent le gradient de ce potentiel, sans tenir compte du paramètre multiplicatif Z^* appliqué aux deux appels à $h(x)$. Entre les deux schémas, le facteur k a été divisé par 10. La forme de la fonction n'est quasiment pas perturbée par ce paramètre.



(a) (b) (c)

FIG. 4.6: Gestion des déplacements le long de l'axe optique : (a) fonction de potentiel, (b) gradient du potentiel pour k = 100, (c) gradient du potentiel pour k = 10

Si l'on considère cette seule mesure de potentiel, la force artificielle associée est de la forme :

$$\overrightarrow{\mathbf{F}_a} = -\epsilon \mathbf{L}_\mathbf{a}^{\|^+} \overrightarrow{\nabla_a} V_a$$

4.2.2.3 Potentiel basé orientation

Un système robotique doté des deux forces décrites précédemment ne peut cependant pas réaliser tous les mouvements. Il reste une ambiguïté pour les situations où le système robotique doit pivoter autour d'un objet. Pour ces configurations, la projection des primitives dans le plan image courant ne permet pas de déterminer si la caméra doit effectuer un mouvement de rotation ou de translation pour rendre visible les points qui ne le sont pas. En effet, comme on l'a déjà dit, si l'on considère que les primitives non visibles se situent au dessus du cadre image, le robot peut aussi bien effectuer une rotation autour de $\overrightarrow{x}$ qu'une translation sur $\overrightarrow{y}$ pour les rendre visibles (voir la figure 4.4).

Pour lever cette ambiguïté, nous considérons que si un mouvement de rotation est observé entre deux prises de vue du chemin, celui-ci doit être au moins partiellement réalisé par le système robotique.

De l'homographie entre l'image courante et la prochaine image du chemin ψ_{i+1} est estimée la rotation entre la pose courante de la caméra et la pose associée à la vue ψ_{i+1}. La représentation vectorielle $\theta\mathbf{u}$ est obtenue de manière unique à partir des coefficients $r_{ij(i=1..3,j=1..3)}$ de la matrice de rotation ${}^{i+1}\mathbf{R}_t$, selon l´équation suivante [Malis 98] :

$$\theta\mathbf{u} = \frac{1}{2\text{sinc}\theta} \begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix},$$

avec $\theta = \arccos((r_{11} + r_{22} + r_{33} - 1)/2)$, et où le sinus cardinal sincθ est tel que $\sin\theta = \theta\text{sinc}\theta$. Notons, dès à présent, que nous ne nous intéressons pas aux rotations autour de l'axe optique. En effet, ces rotations ne permettent pas de faire évoluer le contenu visuel de l'image. Les mouvements réalisés par la caméra ne seront donc pas dépendant des éventuelles rotations autour de cet axe qui pourraient être observées entre les différentes images du chemin.

Finalement, nous voulons éviter que la différence d'orientation mesurée entre l'image courante ψ_t et l'image ψ_{i+1} soit trop importante. Nous pouvons donc définir un intervalle de confiance pour chacune des deux rotations θu_x et θu_y :

$$-p_\theta < \theta u_i < p_\theta,$$

et définir le potentiel suivant pour chacune des deux grandeurs :

$$V_{\theta\mathbf{u}}(\theta u_i) = g(\theta u_i - p_\theta) + g(-p_\theta - \theta u_i). \tag{4.21}$$

Le gradient associé est alors :

$$\overrightarrow{\nabla_{\theta\mathbf{u}}} V_{\theta\mathbf{u}}(\theta u_i) = (\theta u_i - p_\theta)h(\theta u_i - p_\theta) + (\theta u_i + p_\theta)h(-p_\theta - \theta u_i) + \mathcal{O}(\theta u_i - p_\theta) - \mathcal{O}(\theta u_i + p_\theta) \tag{4.22}$$

Les courbes de cette fonction de potentiel et de son gradient sont équivalentes à celles obtenues pour le potentiel précédent. Si l'on considère cette seule mesure de potentiel, la force artificielle associée est de la forme :

$$\overrightarrow{\mathbf{F}}_{\theta\mathbf{u}} = -\epsilon \mathbf{L}_{\theta\mathbf{u}}^{-1} \overrightarrow{\nabla_{\theta\mathbf{u}}}^{\top} V_{\theta\mathbf{u}}, \quad \text{avec} \quad \overrightarrow{\nabla_{\theta\mathbf{u}}}^{\top} V_{\theta\mathbf{u}} = \left(\overrightarrow{\nabla_{\theta u_x}} V_{\theta u_x} , \overrightarrow{\nabla_{\theta u_y}} V_{\theta u_y} \right)$$

et où la matrice d'interaction $\mathbf{L}_{\theta\mathbf{u}}$ est [Malis 98] :

$$\mathbf{L}_{\theta\mathbf{u}} = [\mathbf{0}_3 \;\; \mathbf{L}_w], \tag{4.23}$$

avec :

$$\mathbf{L}_w = \mathbb{I}_3 - \frac{\theta}{2}\tilde{\mathbf{u}} + \left(1 - \frac{\text{sinc}\theta}{\text{sinc}^2\frac{\theta}{2}}\right)\tilde{\mathbf{u}}^2$$

La partie suivante présente comment ces différents potentiels peuvent être considérés conjointement pour définir la loi de commande du système robotique.

4.2.3 Génération de la loi de commande

Généralement, la force appliquée au système robotique est obtenue par une somme pondérée des différentes forces calculées. Cette opération se base sur l'hypothèse que chacune des forces définies permet de contrôler pleinement les mouvements du robot.

Cette hypothèse n'est pas vérifiée pour les champs de potentiel que nous avons définis. En effet les mouvements du robot doivent satisfaire conjointement les trois contraintes posées, et non pas de manière indépendante.

Nous proposons donc un schéma de contrôle prenant en compte simultanément les gradients des trois potentiels. Comme nous l'avons présenté dans la section 4.2.1.2, la force appliquée au robot peut s'écrire sous la forme :

$$\overrightarrow{\mathbf{F}}(\mathbf{p}) = -\epsilon \mathbf{L}^{-1} \overrightarrow{\nabla}^{\top},$$

avec :

$$\overrightarrow{\nabla}^{\top} = \left(\overrightarrow{\nabla_{\mathbf{s}}} V_{\mathbf{s}}^{\top}, \overrightarrow{\nabla_a} V_{a_n}^{\top}, \overrightarrow{\nabla_{\theta\mathbf{u}}} V_{\theta\mathbf{u}}^{\top} \right), \quad \text{et} \quad \mathbf{L} = (\mathbf{L}_{\mathbf{s}}, \mathbf{L}_{a_n}, \mathbf{L}_{\theta\mathbf{u}})$$

Le gradient $\overrightarrow{\nabla_{\mathbf{s}}} V_{\mathbf{s}}^{\top}$ est composé comme présenté en (4.12), page 121. Les primitives ${}^t\mathbf{x}_{n_j}$ considérées forment l'ensemble $\mathcal{M}_i$; elles correspondent à l'ensemble des primitives qui ont été mises en correspondance entre la vue ψ_i et ψ_{i+1}. Les coordonnées des primitives ${}^t\mathbf{x}_{n_j}$ sont soit obtenues par la phase de suivi, soit estimées par projection à partir de la matrice d'homographie.

Dans $\overrightarrow{\nabla_a} V_{a_n}^{\top}$ (définie par l'équation (4.12)), la mesure désirée a^* est calculée en fonction de la projection des primitives $\mathbf{X}_j$ observées dans la vue ψ_{i+1} du chemin. Nous nous limitons encore une fois aux primitives ${}^t\mathbf{x}_{n_j} \in \mathcal{M}_i$. La mesure a est déduite de la position courante des points considérés sur le plan image $\mathcal{I}_t$.

De l'homographie entre la vue courante et la vue ψ_{i+1} est extraite la rotation ${}^{i+1}\mathbf{R}_t$. Sa transposée nous permet de déduire la représentation vectorielle $\theta_t \mathbf{u}_t$ associée, afin de définir à la fois $\overrightarrow{\nabla_{\theta\mathbf{u}}} V_{\theta\mathbf{u}}$ et $\mathbf{L}_{\theta\mathbf{u}}$, suivant les équations (4.22) et (4.23).

Finalement, nous nous retrouvons avec trois ensembles de mesures, afin de contrôler les cinq degrés de liberté de la caméra. La première mesure a pour charge de contrôler principalement les translations sur $\vec{x}$ et $\vec{y}$, la deuxième concerne $\vec{z}$, et la dernière les rotations autour des axes $\vec{x}$ et $\vec{y}$.

Ce schéma de contrôle, prenant conjointement en compte les gradients des trois potentiels, peut être considéré comme un asservissement visuel « qualitatif ». En effet, de la même manière qu'un asservissement visuel, les mouvements du système robotique tendent à faire converger les différentes mesures vers une valeur désirée (dans notre cas, c'est un gradient nul). Le terme *qualitatif* correspond au fait qu'il n'y a pas qu'une seule position du robot qui permet d'atteindre cette convergence. Comme le montrent les figures 4.3 et 4.6, l'utilisation de la fonction arc-tangente normalisée permet d'obtenir, non plus une seule valeur, mais tout un intervalle de mesures où le gradient est nul.

L'intérêt de cette approche est qu'elle permet d'assurer que la position du système robotique vérifie certaines contraintes (comme des conditions d'observabilité satisfaisantes), sans pour autant imposer au robot de suivre une trajectoire prédifinie.

La force appliquée au système robotique est lissée en effectuant un moyennage avec les forces calculées pour les images précédentes (en pratique trois ou quatre mesures de force sont considérées). Ce lissage permet d'atténuer les fortes accélérations qui pourraient apparaître en ne considérant à chaque itération de la boucle de commande que la force calculée sur l'image courante (ces changements étant dûs à l'apparition et à la disparition des primitives).

La figure 4.7 résume les calculs effectués pour déterminer la force à appliquer au système robotique. Nous supposons toujours ici connaître l'ensemble de points $\mathcal{M}_i$ que la caméra doit chercher à posséder dans son champ de vision.

Connaissant la loi de commande appliquée au robot pour un ensemble de points d'intérêt $\mathcal{M}_i$, nous pouvons maintenant nous intéresser au schéma global de contrôle du robot pour une tâche de navigation définie par un ensemble d'images.

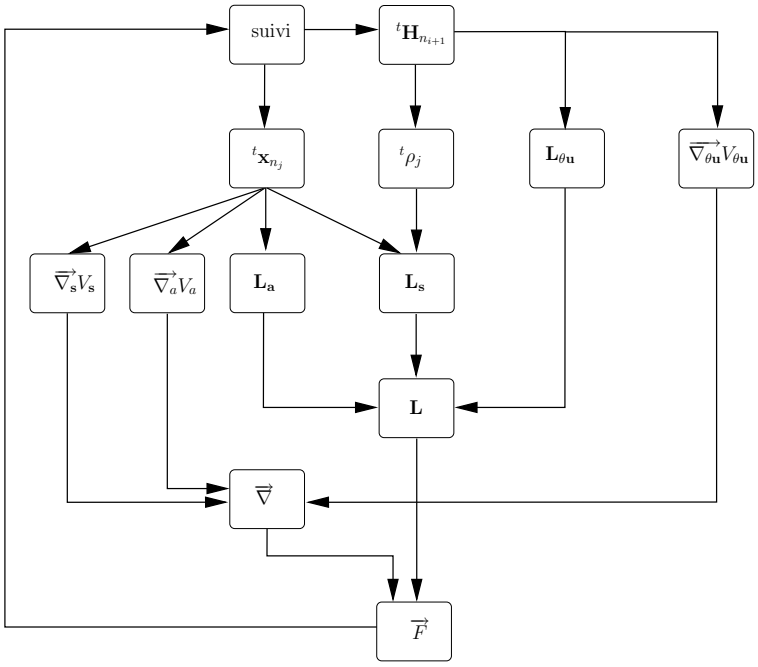


FIG. 4.7: Schéma bloc de contrôle des mouvements. L'ensemble d'intérêt est supposé connu.

4.2.4 Schéma général de navigation

Le schéma de navigation global est présenté sur la figure 4.8. Celui-ci est appliqué en boucle fermée, pour chaque image acquise par la caméra. Dans cette boucle, les informations utilisées sont le chemin d'images, ainsi que la position dans l'image courante de primitives initialement appariées entre deux vues du chemin. Si l'on considère connaître la position de primitives ${}^{t-1}\mathbf{x}_{n_j}$ dans l'image précédente ψ_{t-1}, les opérations successives sont :

1. *Suivi des primitives visibles*, pour obtenir leurs positions ${}^t\mathbf{x}_{n_j}$ dans la nouvelle prise de vue ψ_t.
2. *Détermination des homographies courantes* ${}^t\mathbf{H}_{n_i}$, entre la prise de vue courante ψ_t et les images ψ_i du chemin.
3. *Mise à jour des primitives visibles* : grâce aux homographies précédentes, nous pouvons reprojeter l'ensemble des primitives dans le plan image courant, ce qui permet de détecter si de nouvelles primitives sont visibles. On rappelle que la projection d'une primitive ${}^t\mathbf{x}_{n_j}$ est obtenue à partir de sa position ${}^i\mathbf{x}_{n_j}$ dans une des vues du chemin par la relation :

$$ {}^t\mathbf{x}_{n_j} \propto {}^t\mathbf{H}_{n_i}\,{}^i\mathbf{x}_{n_j} + \beta_j \mathbf{c}_i, $$

   où la parallaxe β_j est nulle si le point appartient au plan de référence de l'homographie.
4. *Sélection des points d'intérêt* : cette étape consiste à choisir l'ensemble de points $\mathcal{M}_i$ que l'on veut faire rentrer dans le champ de vision de la caméra, afin que le robot poursuive son chemin vers sa position désirée.
5. *Calcul de la force attractive* : à partir des positions courantes ${}^t\mathbf{x}_{n_j}$ de ces primitives, de leurs positions ${}^{i+1}\mathbf{x}_{n_j}$ dans la vue ψ_{i+1} et des potentiels $V_{\mathbf{s}}$, V_{a_n} et $V_{\theta\mathbf{u}}$ que nous proposons, nous

déduisons la force à appliquer au robot pour faire rentrer ces points dans le champ de vision de la caméra.

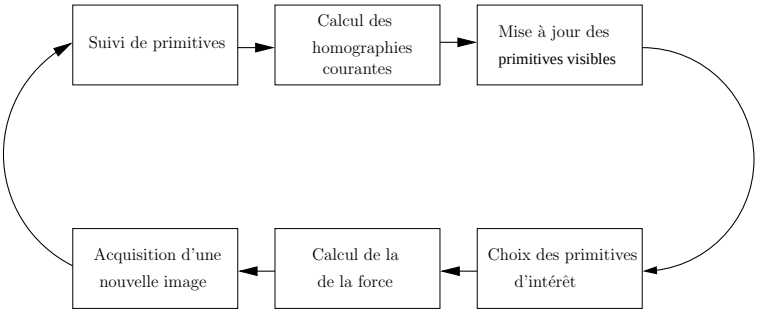


FIG. 4.8: Boucle générale de commande du système robotique

Les trois premières opérations concernent la phase de suivi déjà présentée dans le chapitre précédent. Nous renvoyons le lecteur à ce chapitre pour plus de détails sur ces étapes. Pour plus de clarté, nous effectuons un rappel des informations fournies par la phase de suivi :

- la position sur le plan image courant des primitives du chemin, à savoir ${}^t\mathbf{x}_{n_j}$. Cet ensemble est composé de points visibles dans l'image courante, et de points qui sont hors du champ de vision de la caméra, mais dont les positions sont estimées par reprojection.
- les homographies ${}^t\mathbf{H}_i$ des images du chemin vers l'image courante. Nous rappelons que chacune d'elles est définie par rapport à un plan de la scène, et ne peut donc être estimée que si des points de ce plan sont visibles.

4.2.4.1 Détermination de l'ensemble de points d'intérêt

Le formalisme de choix de l'ensemble d'intérêt découle de l'illustration présentée en début de cette section. Puisqu'un environnement peut être décrit par les primitives qui sont observées par une caméra, les ensembles de points $\mathcal{M}_i$ contenus dans le chemin d'images constituent des descriptions visuelles des différents lieux successifs que doit traverser le système robotique. Ainsi l'ensemble $\mathcal{M}_0$ décrit la portion de l'environnement que la caméra observe avant le début du déplacement. L'ensemble $\mathcal{M}_N$ correspond à l'espace que doit décrire l'image de la caméra une fois la tâche de navigation réalisée.

Les seules connaissances du système robotique sur son environnement correspondent à l'ensemble des primitives détectées dans les images du chemin. Nous devons donc nous assurer que celui-ci conserve toujours dans le champ de vision de sa caméra certaines de ces primitives. De ce fait, nous ne pouvons considérer comme ensemble d'intérêt un ensemble dont aucune primitive n'est actuellement visible. Rien ne permet d'assurer en effet qu'il existe un chemin direct vers cette zone permettant d'assurer la contrainte de visibilité. L'ajout de potentiels répulsifs pour éviter que les points visibles ne disparaissent durant ce mouvement risquerait de générer de nombreux minima locaux dans lesquels pourrait rester bloqué le système robotique.

C'est pourquoi la sélection de l'ensemble d'intérêt ne se fait que parmi ceux dont certaines primitives sont déjà visibles dans l'image fournie par la caméra. Plus précisément, les points vi-

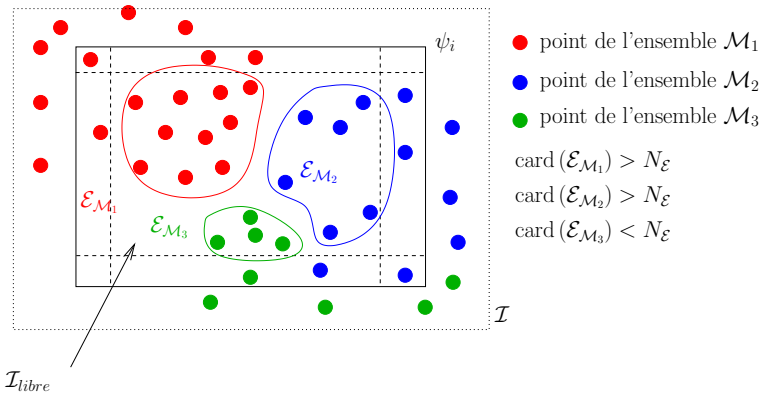


FIG. 4.9: Sélection de l'ensemble d'intérêt pour la navigation

sibles sont regroupés par rapport à l'ensemble $\mathcal{M}_i$ dont ils proviennent. On note $\mathcal{E}_i$ l'ensemble des primitives de $\mathcal{M}_i$ qui sont actuellement dans le champ de vision de la caméra (voir la figure 4.9).

Puisque nous voulons nous asservir sur un lieu proche, nous ne considérons que les ensembles $\mathcal{E}_{\mathcal{M}_i}$ tels que :

$$\text{card}\,(\mathcal{E}_{\mathcal{M}_i}) > N_\mathcal{E},$$

où card(E) est le cardinal de l'ensemble E, et $N_\mathcal{E}$ un seuil. De plus, pour pouvoir correctement détecter les primitives du lieu d'intérêt, il est nécessaire de connaître la matrice d'homographie permettant de relier ce lieu à l'image courante. Cette contrainte nous permet de nous assurer que tout en se déplaçant vers ce nouveau lieu, le système robotique aura la possibilité de localiser les primitives le définissant. Nous rajoutons donc une autre contrainte pour la sélection de l'ensemble d'intérêt, que l'on nomme $Base?(\mathcal{M}_i)$, prédicat qui spécifie si l'homographie reliant le lieu i avec la vue courante est à jour.

Finalement, parmi tous les ensembles de points $\mathcal{M}_i$ du chemin satisfaisant :

$$\text{card}\,(\mathcal{E}_{\mathcal{M}_i}) > N_\mathcal{E} \quad \text{et} \quad Base?(\mathcal{M}_i),$$

nous choisissons celui d'indice i supérieur sur le chemin. La position des primitives visibles de cet ensemble, et l'estimation de celles qui se projettent hors du champ de vision de la caméra constituent l'ensemble de primitives à partir duquel sont contrôlés les mouvements du robot.

La boucle présentée sur la figure 4.8 est appliquée jusqu'à ce que des primitives décrivant le dernier lieu à traverser, à savoir $\mathcal{M}_N$, soient visibles. La section suivante traite de la loi de commande à appliquer lorsque le robot est proche de sa position désirée.

4.2.4.2 Convergence vers la position désirée

Lorsque des points du dernier ensemble sont visibles, la tâche de navigation devient une tâche de positionnement qui peut être effectuée par une technique standard d'asservissement visuel. Si l'on estime être loin de la position désirée, nous risquons d'être aux limites de la zone de convergence d'un asservissement visuel $2D$. Il est donc préférable de se rapprocher de la position désirée par un formalisme possédant un plus grand espace de convergence. Nous proposons donc,

dans un premier temps, de contrôler les mouvements du robot par un potentiel attractif défini à partir de l'estimation de la pose partielle du robot.

La force attractive est définie à partir de l'estimation de la pose partielle $\mathbf{p} = [\mathbf{t}_{dk} \;\; \theta\mathbf{u}]$ entre la position courante de la caméra, et la position associée à la dernière image du chemin. Nous rappelons que la translation à un facteur d'échelle près ainsi que la rotation entre deux prises de vue peuvent être extraites de la matrice d'homographie ${}^{t}\mathbf{H}_{n_N}$ reliant les projetés de ces deux images. Si l'on modélise le potentiel attractif par une fonction parabolique :

$$V_{\mathbf{p}} = \frac{1}{2}\|\mathbf{p_t}\|^2,$$

alors la force attractive est de la forme :

$$\overrightarrow{\mathbf{F}_{\mathbf{p}}} = -\epsilon \mathbf{L}_{\mathbf{p}}^{+} \overrightarrow{\nabla_{\mathbf{p}}}^{\top} V_{\mathbf{p}},$$

où $\overrightarrow{\nabla_{\mathbf{p}}}^{\top} V_{\mathbf{p}} = \mathbf{p}$. La matrice d'interaction associée est la suivante :

$$\mathbf{L}_{\mathbf{p}} = \begin{bmatrix} \mathbb{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_w \end{bmatrix}$$

En s'appuyant sur le fait que $\mathbf{L}_w^{-1}\theta\mathbf{u} = \theta\mathbf{u}$ [Malis 98], nous pouvons donc déduire que la force attractive peut s'écrire :

$$\overrightarrow{\mathbf{F}_{\mathbf{p}}} = -\epsilon \mathbf{p}(t)$$

Notons que cette loi de commande correspond exactement à un asservissement visuel hybride comme présenté dans [Malis 02].

Durant cette phase de déplacement, l'estimation de l'homographie ${}^{t}\mathbf{H}_{n_N}$ permet non seulement de mettre à jour l'estimée de la pose de la caméra $\mathbf{p}_t$, mais aussi de détecter l'apparition de nouvelles primitives initialement détectées dans la dernière image. La phase de positionnement se finit lorsque l'erreur mesurée entre les positions courante et désirée des primitives du dernier ensemble est inférieure à un seuil donné.

Une fois que la mesure d'erreur entre les positions courante ${}^{t}\mathbf{x}_{n_j}$ et désirée ${}^{N}\mathbf{x}_{n_j}$ est suffisamment faible, il est possible de finir la tâche de navigation en effectuant un asservissement visuel $2D$ classique, afin d'améliorer la précision du positionnement (attendre une erreur $({}^{t}\mathbf{x}_{n_j} - {}^{N}\mathbf{x}_{n_j})$ faible avant de réaliser l'asservissement visuel $2D$ permet de s'assurer que le système ne risque pas de diverger).

4.3 Résultats expérimentaux

Dans cette section, nous présentons un ensemble d'expériences réalisées sur ce formalisme. Nous avons considéré plusieurs types de systèmes robotiques, possédant plus ou moins de degrés de liberté. Notons, dès à présent, que toutes les expériences n'ont pas été réalisées avec des systèmes robotiques réels. L'utilisation d'un simulateur dans la plupart de nos expériences nous a permis de pouvoir étudier la validité théorique de notre schéma de navigation en s'affranchissant du bruit potentiel sur les mesures images que pourrait provoquer une phase de suivi de primitives.

4.3.1 Cas d'un environnement de navigation plan

La première expérience que nous présentons concerne le contrôle des mouvements d'un bras articulaire possédant une caméra embarquée sur son effecteur. La scène observée est plane. Le chemin d'images permettant de relier l'image fournie par la caméra avant le déplacement et celle que doit fournir cette même caméra une fois la navigation réalisée est présentée sur la figure 4.10. La figure 4.11 présente une projection sur le premier plan de l'ensemble des primitives mises en correspondance sur le chemin d'images. Cette figure montre qu'il est impossible de se diriger directement de la position initiale vers la position désirée. La zone entre ces deux images n'est en effet décrite par aucune primitive. Elle constitue en somme une zone interdite pour le robot.

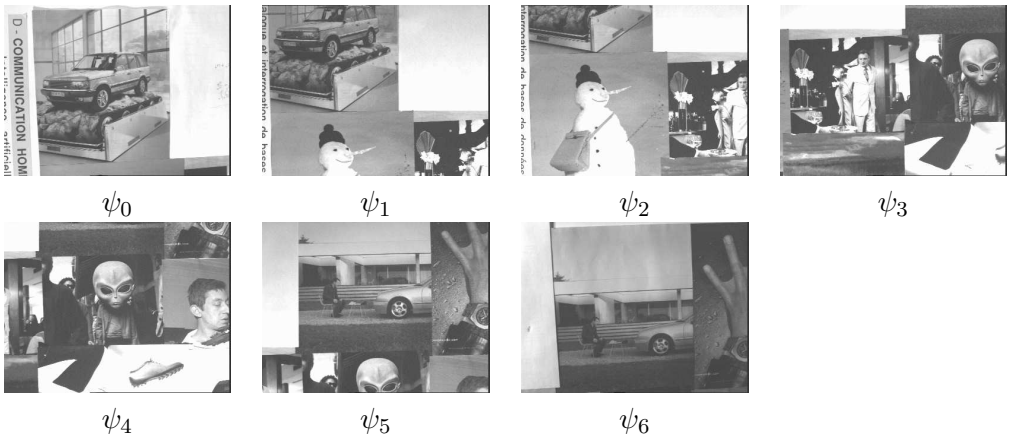


FIG. 4.10: Chemin d'images reliant les vues initiale (ψ_0) et désirée (ψ_6)

La figure 4.12 présente la trajectoire du point principal de la caméra durant le déplacement. On peut observer que les positions associées aux images du chemin ne sont pas atteintes. La figure 4.13 présente une comparaison de la trajectoire obtenue par le formalisme que nous proposons avec deux autres méthodes de navigation effectuant une convergence vers les différentes images intermédiaires [Remazeilles 02]. La première méthode correspond à un ensemble d'asservissements visuels $2D$ successifs. Le système robotique converge totalement vers chacune des vues du chemin. Une fois que l'erreur mesurée entre les positions courante et désirée des primitives d'intérêt est inférieure à un certain seuil, le système robotique considère l'image suivante comme une nouvelle position désirée à atteindre.

La deuxième technique consiste de la même manière à converger vers les différentes positions intermédiaires, mais elle n'attend par la convergence totale vers chacune des images du chemin. Le calcul des matrices d'homographie ${}^t\mathbf{H}_{n_i}$ entre la vue courante et les différentes vues du chemin permet de décider quand le système peut changer d'images et de primitives d'intérêts. Si le système converge vers la vue ψ_i du chemin, l'homographie ${}^t\mathbf{H}_i$ permet de projeter sur le plan image courant les primitives en correspondance entre les vues ψ_i et ψ_{i+1}. Lorsque suffisamment de primitives sont visibles pour assurer un asservissement visuel $2D$, le système arrête son asservissement courant pour s'asservir sur la prochaine image ψ_{i+1} du chemin (le formalisme d'estimation de la matrice d'homographie est analogue à celui présenté dans le chapitre 2 pour les environnements plans).

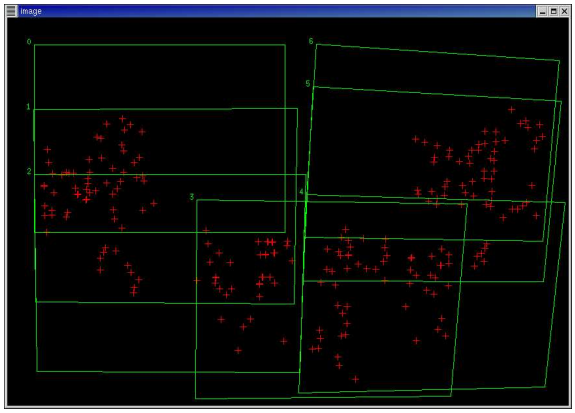


FIG. 4.11: Projection de l'ensemble des primitives et des cadres images des vues du chemin sur le premier plan image

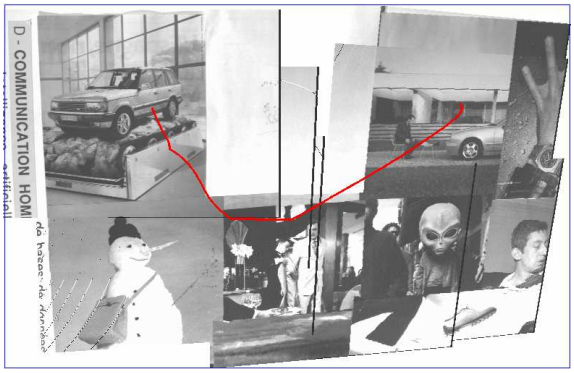


FIG. 4.12: Trajectoire du point principal projetée sur le premier plan image.

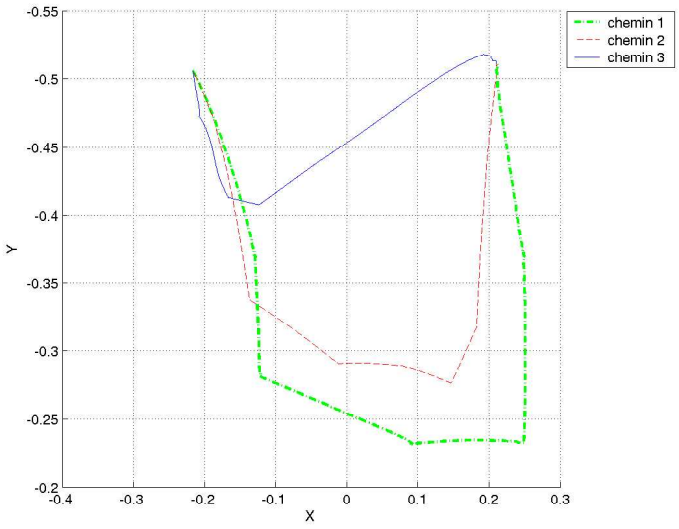


FIG. 4.13: Comparaison de la trajectoire obtenue (en bleu) avec celles réalisées suivant deux autres techniques de navigation.

Le formalisme proposé, tout en assurant de conserver le système robotique dans une zone où suffisamment de primitives du chemin restent visibles par la caméra, permet d'obtenir une meilleure trajectoire en terme de distance parcourue. Si les trajectoires des deux autres méthodes montrent clairement leurs dépendances envers les différentes positions intermédiaires, cette dépendance est beaucoup moins explicite sur la trajectoire obtenue par notre méthode.

Pour montrer l'indépendance aux positions des images intermédiaires, une rotation de 180 degrés a été appliqué aux images ψ_2 et ψ_5 du chemin (le chemin est alors celui de la figure 4.14). La résolution de ce chemin par des asservissements visuel $2D$ successifs contraindrait le robot à effectuer des rotations inutiles lors des déplacements $\psi_1 - \psi_2$, $\psi_2 - \psi_3$, $\psi_4 - \psi_5$ et $\psi_5 - \psi_6$. La deuxième méthode présentée ci-dessus, bien qu'elle permette de ne pas converger vers les images intermédiaires, réaliserait également une partie de ces rotations.

La figure 4.15 compare la trajectoire associée au chemin sans rotation présenté en 4.10, avec celle obtenue lorsque les images b et f sont retournées. Ces deux trajectoires sont quasiment identiques, ce qui prouve bien que la méthode est indépendante des positions des prises de vue.

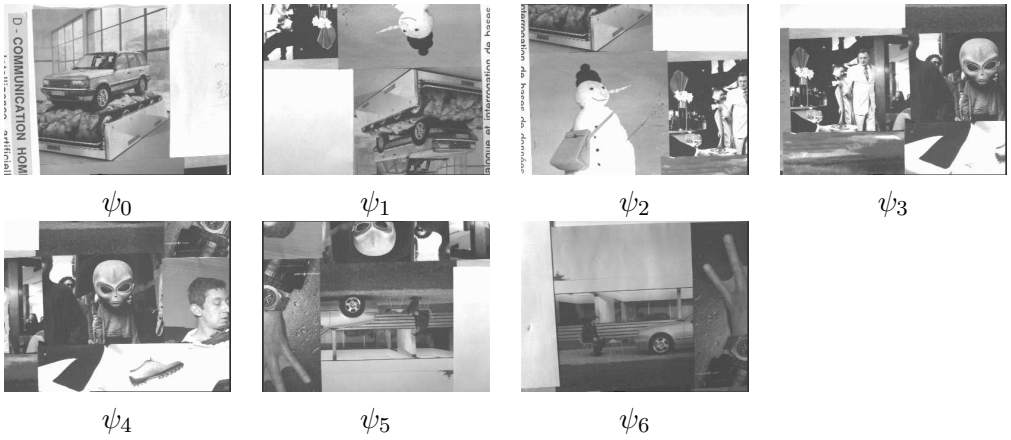


FIG. 4.14: Images constituant le chemin à parcourir, avec deux images inversées (ψ_0 : position initiale, ψ_6 : position finale)

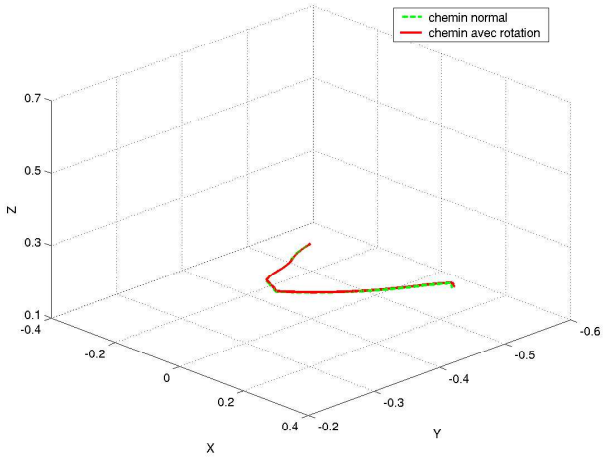


FIG. 4.15: Comparaison de la trajectoire du robot réalisée pour les chemins des figures 4.10 et 4.14

4.3.2 Cas d'un système robotique à cinq degrés de liberté

Les résultats présentés ci-dessous ont été obtenus en simulation. L'objet autour duquel doit se déplacer la caméra est présenté sur la figure 4.16. Il est composé d'un ensemble de plans. À ces différents plans sont associés un certains nombre de points (la figure 4.17 présente certains de ces points). Les points de la scène ne sont pas tous situés sur les différents plans de la scène. Notons de plus que dans le simulateur le problème des faces partiellement occultées n'est pas gérée (comme on peut le voir sur la vue de droite de la figure 4.16.

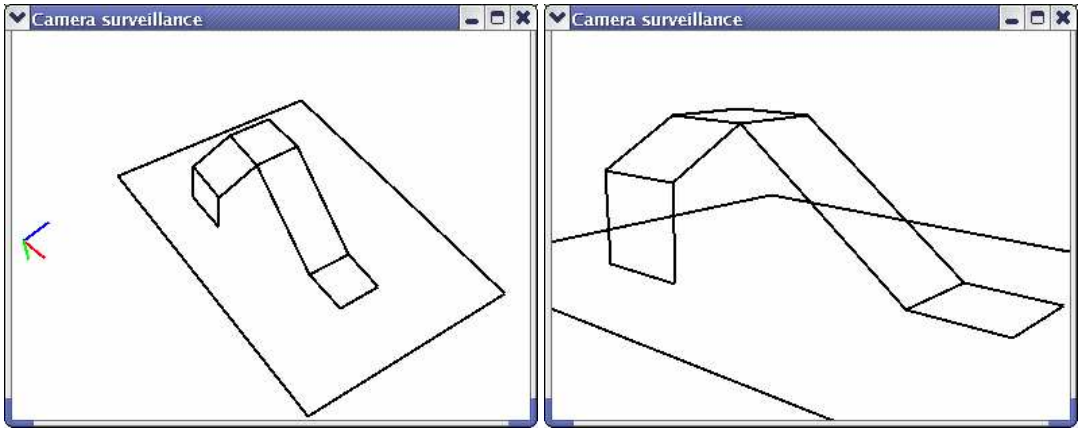


FIG. 4.16: Objet constituant la scène tridimensionnelle pour la simulation

La figure 4.17 présente une tâche de navigation. Plus exactement, elle donne les ensembles de primitives $\mathcal{M}_i$ (les points en correspondance entre chaque couple d'images successives du chemin sont tracées en rouge. Les autres points visibles mais non appariés sont en noir, tout comme les contours des faces de l'objet). Le déplacement revient à se déplacer le long des trois premiers plans de l'objet. Les ensembles $\mathcal{M}_2$, $\mathcal{M}_3$ et $\mathcal{M}_4$ traduisent le mouvement permettant de

se positionner parallèlement au deuxième plan de l'objet. De la même manière, les ensembles $\mathcal{M}_5$ et $\mathcal{M}_6$ concernent le positionnement parallèle au dessus du troisième plan de l'objet.

La figure 4.18 présente une visualisation des positions de la caméra associées à chacune de ces prises de vues. Les caméras sont représentées par leur repères respectifs. Les couleurs rouge, vert et bleu correspondent aux orientations des axes $\vec{x}$, $\vec{y}$ et $\vec{z}$ de ces repères. Les positions des caméras associées aux images initiale et désirée par rapport à un repère caméra référence sont les suivantes (les mesures sont en mètre) :

$$\mathbf{T}_0 = \begin{bmatrix} 1 & 0 & 0 & 0,08 \\ 0 & 1 & 0 & 0.20 \\ 0 & 0 & 1 & -0.80 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad , \quad \mathbf{T}_N = \begin{bmatrix} 1 & 0 & 0 & 0,04 \\ 0 & 0 & 1 & -3.90 \\ 0 & -1 & 0 & 2.54 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Le changement d'ensemble d'intérêt peut entraîner une forte variation de l'accélération de la caméra. La contribution de la force d'un nouvel ensemble de primitives d'intérêt est donc répartie sur plusieurs itérations. Si l'on note $\overrightarrow{\mathbf{F}_{t-1}}$ la force calculée pour l'image précédente ψ_t et $\overrightarrow{\mathbf{F}_i}$ la force calculée pour l'image courante ψ_t $\mathcal{M}_i$, la force $\overrightarrow{\mathbf{F}_t}$ appliquée au robot est alors :

$$\overrightarrow{\mathbf{F}_t} = (1-\alpha)\overrightarrow{\mathbf{F}_{t-1}} + \alpha\overrightarrow{\mathbf{F}_i}$$

Ce lissage des transitions s'effectue uniquement lors des changements d'ensemble d'intérêt. Dans nos expériences, il s'effectue sur une dizaine d'itérations. À chacune de ces itérations, le terme α (qui vaut 0 au changement d'ensemble) est incrémenté de 0.1.

La force appliquée au système robotique est représentée sur la figure 4.20. Les lignes verticales dénotent les changements d'ensemble d'intérêt (les deux derniers ensembles correspondant à la tâche de positionnement comme présentée en 4.2.4.2). Peu de mouvements sont générés suivant $\vec{x}$ et autour de $\vec{y}$, ce qui est cohérent avec la tâche de navigation considérée, qui n'est pas censée mettre à contribution ces degrés de liberté.

Il peut être observé que les changements d'ensemble s'effectuent alors que la vitesse du robot n'est pas nulle. Ce comportement est satisfaisant puisque le robot n'est pas censé converger vers ces différentes zones intermédiaires, mais seulement les traverser.

De plus, les changements d'ensemble d'intérêt se traduisent par une variation de la force appliquée. Même si celle-ci varie fortement dans ces moments de transition (comme entre les étapes 1 et 2 ou 7 et 8 en ce qui concerne les translation sur $\vec{y}$, ou entre 2 et 3 pour les rotations autour de $\vec{y}$), il peut être noté que ces transitions sont étalées sur plusieurs tours de la boucle, grâce au lissage des transitions.

La figure 4.19 présente la trajectoire réalisée par la caméra dans l'espace de travail. Si nous la comparons avec la figure 4.18, nous pouvons vérifier que le système robotique ne converge pas vers chacune des positions intermédiaires du chemin. Les caméras traçées sur la figure 4.19 sont espacées d'un nombre constant d'itération. Il est intéressant de noter que la caméra commence lentement sa tâche de navigation (puisque les premières caméras tracées sont très proches les unes des autres). La vitesse de déplacement augmente donc progressivement, ce qui est un comportement satisfaisant. Enfin, la trajectoire réalisée est relativement lisse et continue, contrairement à celle implicitement proposée par le chemin d'images.

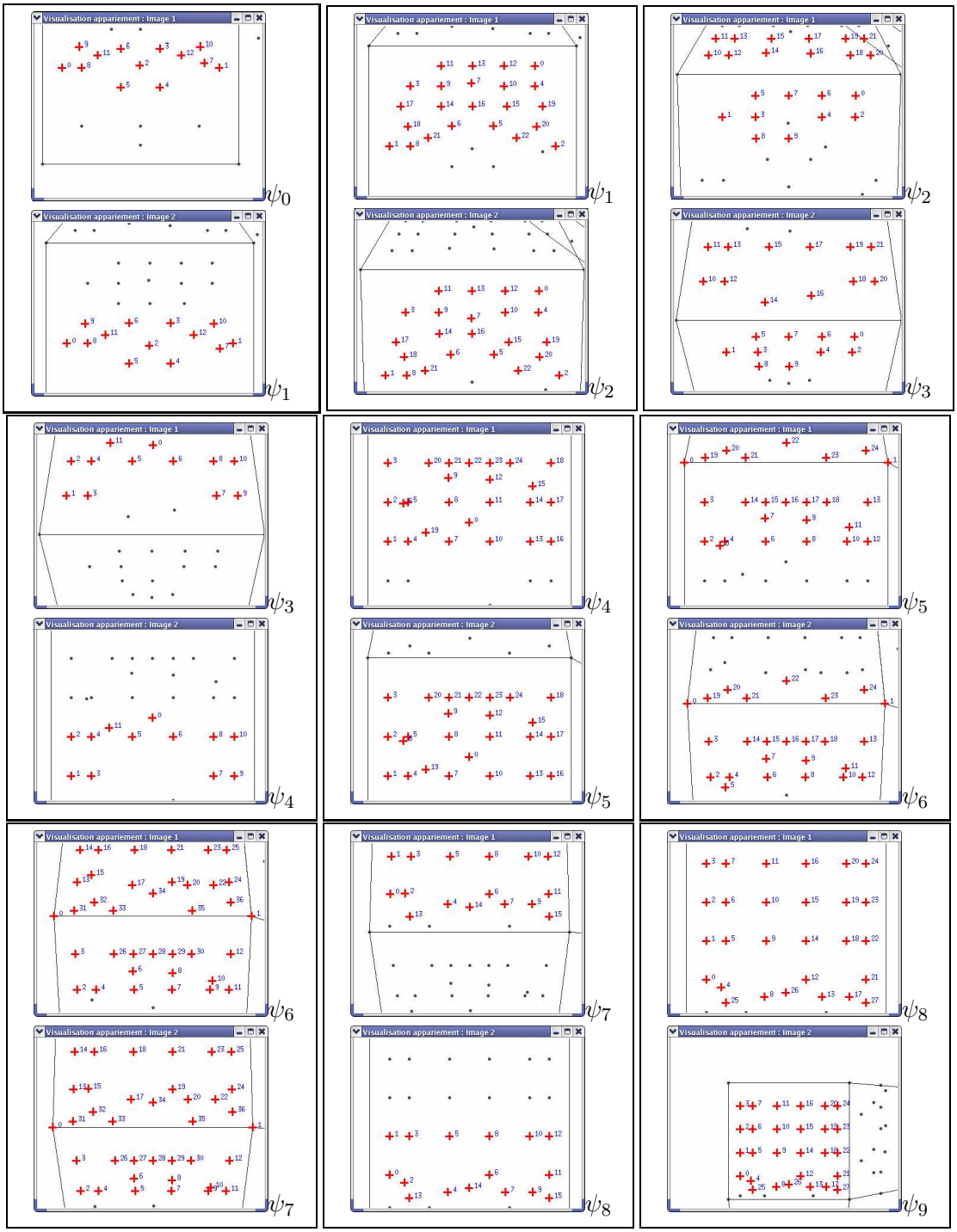


FIG. 4.17: Cas 1 : ensembles de points $\mathcal{M}_i$ définissant l'environnement à parcourir (ψ_0 : position initiale, ψ_9 : position finale). Les points de la scènes sont représentés en noir. Pour chaque couple d'images (ψ_i ψ_{i+1}), les points appariés sont en rouge.

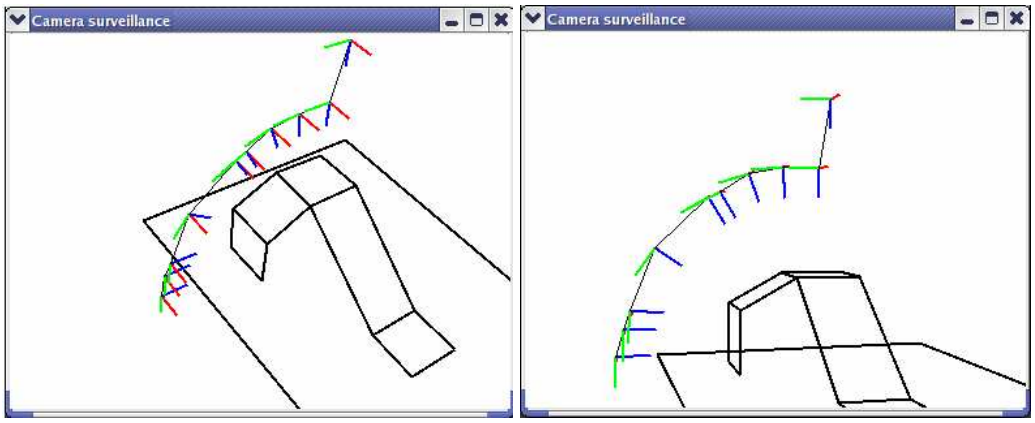


FIG. 4.18: Cas 1 : Position de prise de vue des images du chemin

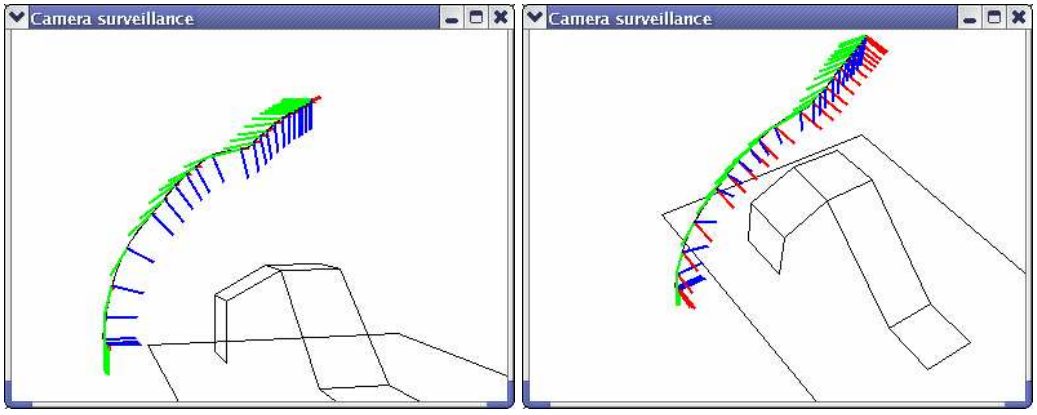


FIG. 4.19: Cas 1 : Visualisation de la trajectoire réalisée

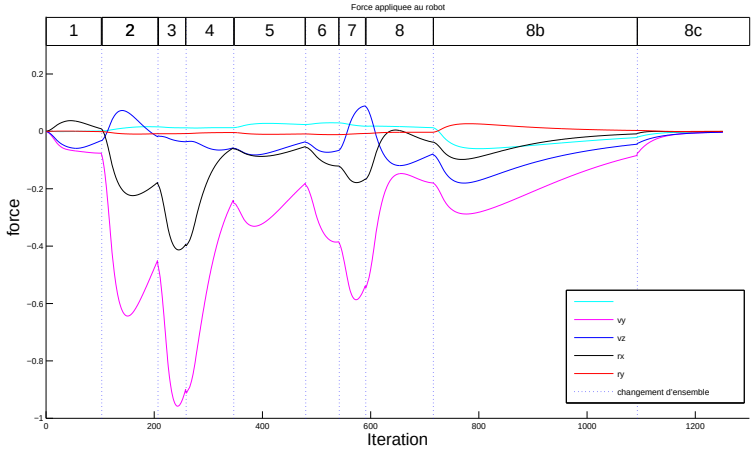


FIG. 4.20: Cas 1 : forces appliquées au système robotique

La figure 4.21 démontre que l'intérêt de la fonction arc-tangente utilisée dans nos fonctions de potentiels (équations (4.10), (4.18) et (4.21)) n'est pas simplement de fournir d'un point de vue théorique une fonction de potentiel continue et dérivable sur l'ensemble de l'espace image. Pour cette expérience, le potentiel V_{a_n} est simplement défini par une fonction continue par morceaux, à savoir :

$$V_{a_n} = \begin{cases} \frac{1}{2}\left(\sqrt{\frac{a^*}{a}} - 1 - p\right)^2 & \text{si} \quad a_n < a_n^*(1-p) \\ \frac{1}{2}\left(\sqrt{\frac{a^*}{a}} - 1 + p\right)^2 & \text{si} \quad a_n > a_n^*(1+p) \\ 0 & \text{sinon} \end{cases}$$

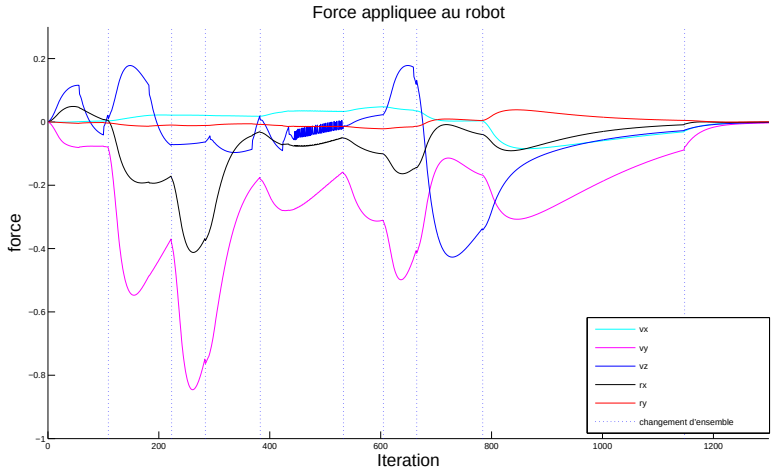


FIG. 4.21: Cas 1 : force appliquée au système robotique (avec une fonction continue par morceaux pour contrôler les mouvements le long de l'axe optique)

Les oscillations observées sur les vitesses de déplacement sur v_z sont dues aux changements de palier de la fonction de potentiel V_{a_n}. Ces oscillations ne sont pas observées sur la figure 4.20, puisque la transition entre les valeurs de potentiel nuls et non nuls s'effectue de manière continue.

La deuxième expérience (figure 4.22) s'intéresse à une trajectoire plus longue, et faisant intervenir tous les degrés de liberté du système robotique. Pour ce faire, la caméra ne se contente pas de monter au dessus de l'objet comme dans l'expérience précédente. Elle doit ensuite se déplacer pour aller observer un plan situé sur le sol. La figure 4.22 présente l'ensemble des images ψ_i permettant de définir cette tâche de navigation.

La figure 4.23 présente les positions associées aux différentes vues de ce chemin. Les positions des caméras associées aux images initiale et désirée par rapport à un repère de référence sont les suivantes :

$$\mathbf{T}_0 = \begin{bmatrix} 1 & 0 & 0 & 0.08 \\ 0 & 1 & 0 & 0.2 \\ 0 & 0 & 1 & -5.2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{T}_N = \begin{bmatrix} 1 & 0 & 0 & 2.98 \\ 0 & 1 & 0 & -2.07 \\ 0 & 0 & 1 & 2.35 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

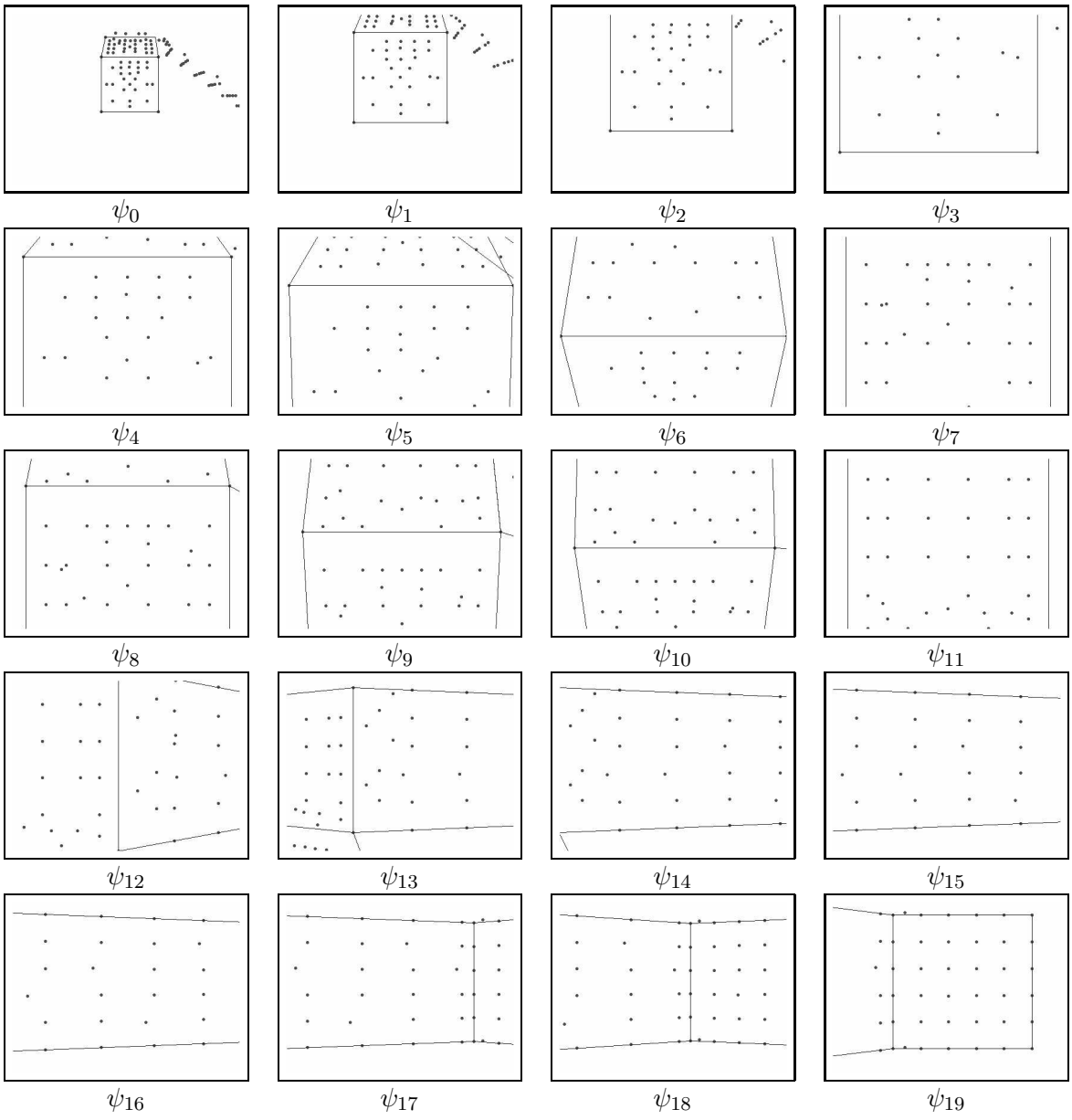


FIG. 4.22: Cas 2 : ensembles d'images définissant l'environnement à parcourir (ψ_0 : position initiale, ψ_{19} : position finale)

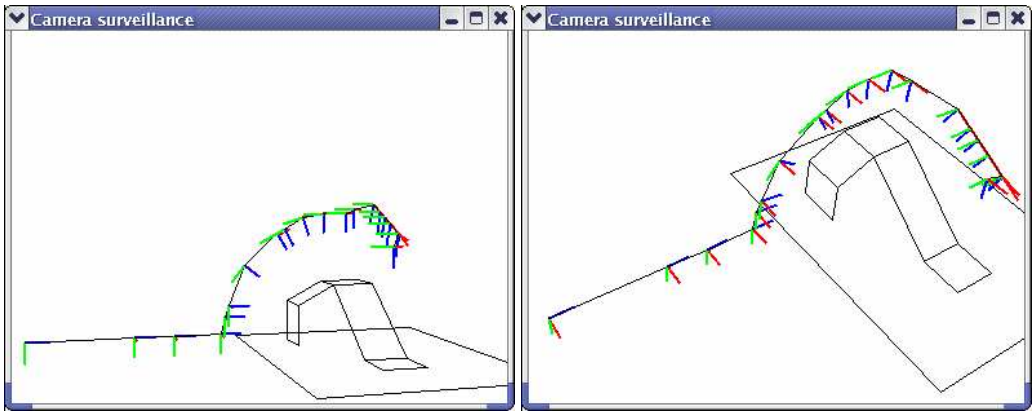


FIG. 4.23: Cas 2 : Positions de prise de vue des images du chemin

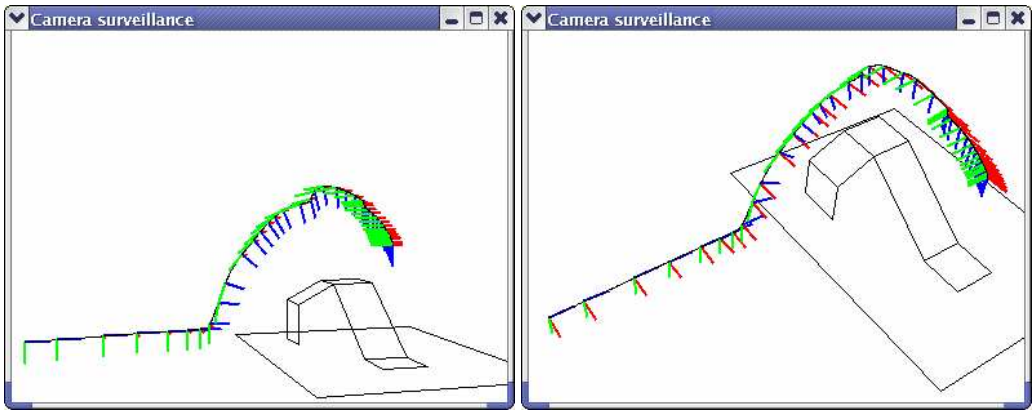


FIG. 4.24: Cas 2 : Visualisation de la trajectoire réalisée

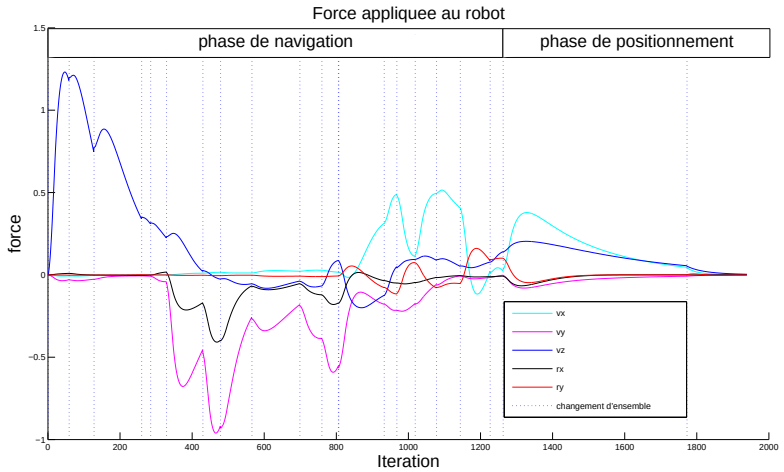


FIG. 4.25: Cas 2 : forces appliquées au système robotique

Les figures 4.24 et 4.25 présentent respectivement la trajectoire réalisée par le système robotique durant sa navigation et la force appliquée au robot. Encore une fois, la trajectoire obtenue, même si elle dépend implicitement des différentes vues du chemin, ne correspond pas explicitement à une convergence vers les différentes positions intermédiaires associées.

Dans le dernier cas présenté, seule la position initiale a été modifiée. La figure 4.26 correspond à l'image associée. La pose de la caméra par rapport au repère de référence est alors :

$$\mathbf{T}_0 = \begin{bmatrix} 1 & 0 & 0 & 1.38 \\ 0 & 1 & 0 & 0.2 \\ 0 & 0 & 1 & -5.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Au début de la navigation, le déplacement est, comme dans l'expérience précédente, composé principalement d'une translation suivant l'axe optique. Durant ce déplacement, les primitives d'intérêt se rapprochent du bord gauche de l'image. Nous observons alors sur les figures 4.27 et 4.29 qu'un mouvement selon l'axe $\vec{x}$ est généré afin de conserver ces primitives dans le champ de vision de la caméra.

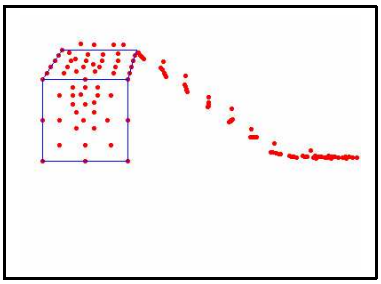


FIG. 4.26: Cas 3 : image fournie par la caméra à la position initiale

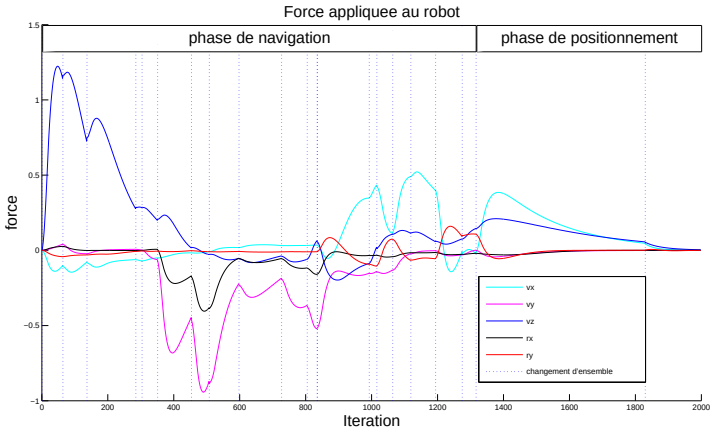


FIG. 4.27: Cas 3 : forces appliquées au système robotique

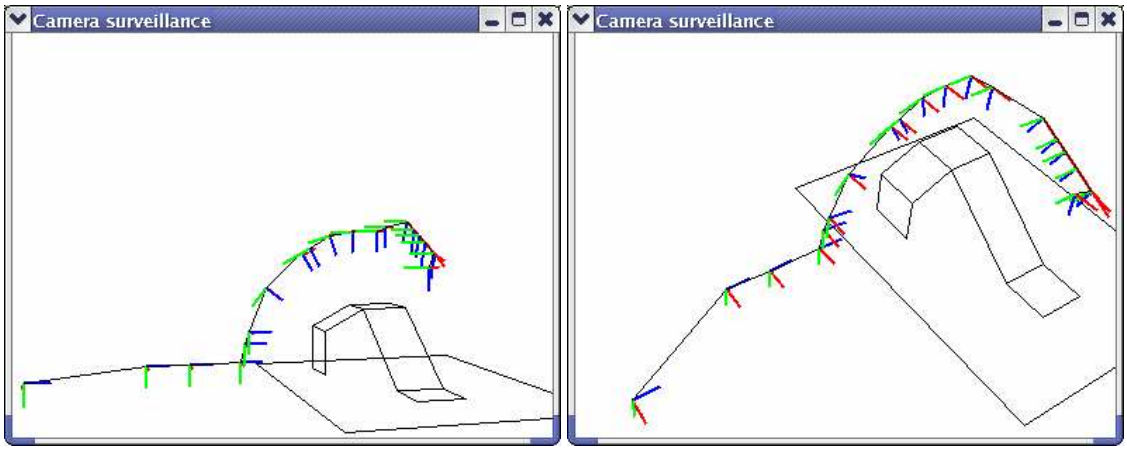


FIG. 4.28: Cas 3 : Positions de prise de vue des images du chemin

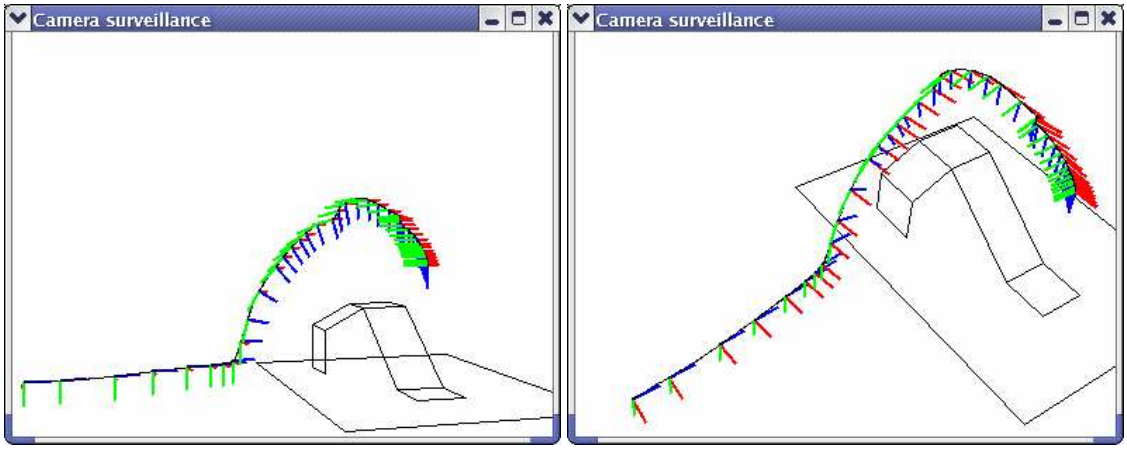


FIG. 4.29: Cas 3 : Visualisation de la trajectoire réalisée

4.3.3 Cas d'un système robotique se déplaçant sur un plan

Les expériences ci-dessous concernent le contrôle des mouvements d'un robot se déplaçant sur un plan. Pour ce type de mouvement, nous avons modélisé le système robotique par un système possédant deux puis trois degrés de liberté. L'environnement de navigation peut être assimilé à un couloir. Des points sont définis dans le voisinage des plans constituant le sol et les murs du couloir. Les tâches de navigation considérées s'intéressent principalement à la réalisation d'un virage par la caméra.

Considérons tout d'abord un système robotique possédant deux entrées de commande. Les mouvements possibles d'un tel système correspondent à des translations suivant l'axe optique $\vec{z}$, ainsi que des rotations autour de l'axe $\vec{y}$. De ce fait, les matrices d'interaction L_s et $L_{\theta \mathbf{u}}$ sont simplifiées pour ne considérer que ces mouvements possibles.

La figure 4.30 présente les positions associées aux différentes images constituant le chemin d'images. Les poses correspondant aux images initiale et désirée sont les suivantes :

$$\mathbf{T}_0 = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1.9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{T}_N = \begin{bmatrix} 0 & 0 & 1 & 10.88 \\ 0 & 1 & 0 & -1 \\ -1 & 0 & 0 & 10.17 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Les figures 4.31 et 4.32 décrivent la trajectoire réalisée par le système robotique. Notons que durant la phase de positionnement finale, des mouvements de translation sont généré suivant l'axe $\vec{x}$, alors que le robot n'est pas censé pouvoir générer des mouvements suivant cet axe. Durant la phase de positionnement, et simplement durant cette phase, nous avons considéré que la caméra avait 3 degrés de liberté (en effet nous n'avons pas traité le cas des robots non holonomes durant ces travaux).

Nous observons sur ces deux schémas que le virage est correctement effectué par le système robotique. Nous observons que les caméras sont plus regroupées durant le virage. Ce mouvement entraîne donc un ralentissement du véhicule. Ce comportement est satisfaisant : un virage à 90 degrés comme celui-ci se traduit par un changement relativement rapide de l'ensemble des primitives visibles. Le fait de ralentir durant ce mouvement permetrait dans une application réelle de pouvoir plus facilement mettre à jour les primitives visibles.

Des oscillations peuvent être observées sur la vitesse de déplacement le long de l'axe $\vec{z}$, vers la fin de la phase de navigation. La caméra se situe alors dans le deuxième couloir. Ce dernier est décrit par des images dont les positions associées sont plus éloignées les unes des autres qu'elles ne l'étaient dans le premier couloir. La mesure de potentiel associée est donc plus grande, ce qui entraîne ces plus grandes variations de vitesse.

L'expérience suivante, où le système robotique possède trois degrés de liberté permet de détecter les limites de validité de notre formalisme. Le troisième degré de liberté rajouté au système de navigation correspond aux translations suivant l'axe $\vec{x}$. La tâche de navigation considérée est la même que celle de l'expérience précédente.

La figure 4.33 présente la trajectoire réalisée par le système robotique. Lorsque celui-ci cherche à faire rentrer dans son champ de vision des primitives situées sur le mur de droite, il génère un mouvement qui le fait sortir du couloir. En effet dans notre formalisme rien ne permet actuellement de prendre en compte les contraintes liées à une délimitation de l'espace libre du système robotique. Nous observons de plus que le système reste en dehors du couloir et ne vient pas converger vers la position désirée. Cela est dû au fait que le simulateur que nous utilisons gère l'occultation des primitives par les plans de la scène. Les points étant de l'autre coté du mur, ils ne sont pas, à juste titre, considérés comme visibles.

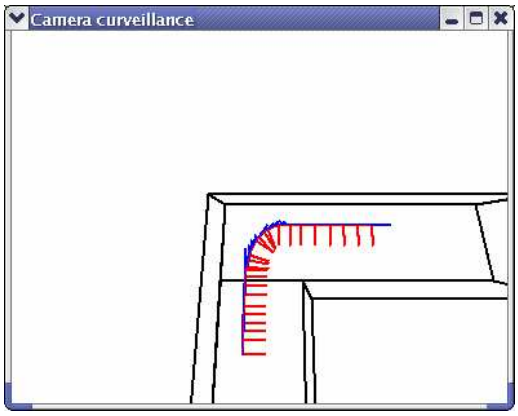


FIG. 4.30: Position de prise de vue des images du chemin

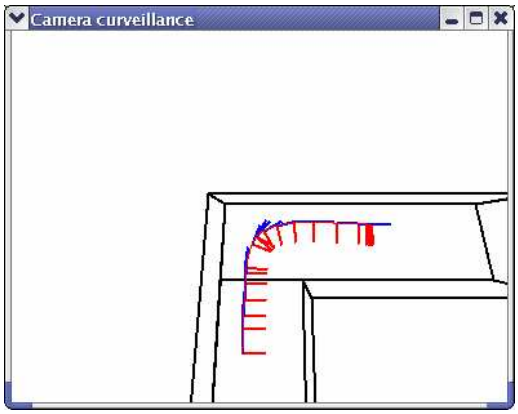


FIG. 4.31: Cas 4 (deux entrées de commande) : trajectoire réalisée par le système robotique

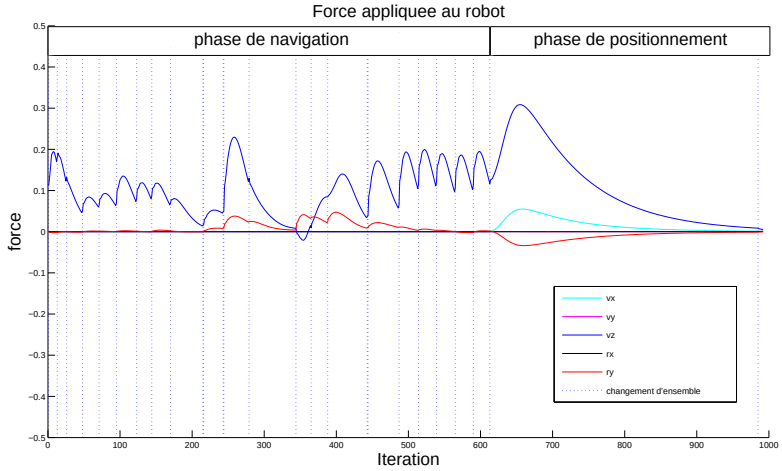


FIG. 4.32: Cas 4 : Forces appliquées au système robotique

Une solution *ad hoc* permet cependant de réaliser ce mouvement. Pour ce faire, nous considérons la rotation entre la position courante de la caméra et la vue ψ_{i+1} associée à l'ensemble d'intérêt courant $\mathcal{M}_i$. La méthode de calcul de la force à appliquer au système robotique dépend alors de la valeur de l'angle de rotation θ_y autour de $\vec{y}$:

$$\overrightarrow{\mathbf{F}} = \begin{cases} -\epsilon \mathbf{L}^{+} \overrightarrow{\nabla} & \text{si } \theta_y < \theta_M \\ -\epsilon \mathbf{L}^{+}_{\theta_\mathbf{y}} \overrightarrow{\nabla_{\theta_y \mathbf{u}}} & \text{sinon,} \end{cases}$$

où θ_M correspond à un angle de rotation maximal. Le problème rencontré dans l'expérience apparaît lorsque la caméra est censée effectuer le virage. En effet celle-ci semble tourner trop lentement. De plus, les déplacements générées sur les axes $\vec{x}$ et $\vec{z}$ ont tendance à limiter la part des rotations dans le mouvement global. C'est pourquoi nous proposons de considérer uniquement le potentiel associée aux rotations lorsque la différence d'orientation θ_y est trop importante pour déterminer le mouvement.

Les figures 4.35 et 4.34 montrent que cette nouvelle loi de commande permet de réaliser correctement la tâche de navigation spécifiée. Cependant, la solution proposée ci-dessus ne nous semble pas satisfaisante d'un point de vue théorique, puisqu'elle ne permet pas d'assurer que le système robotique reste toujours dans la zone libre délimitée dans ces expériences par les murs verticaux. Une solution plus élégante pour traiter ce genre de situation constitue l'une des perspectives de recherche que nous envisageons dans la conclusion de ce chapitre.

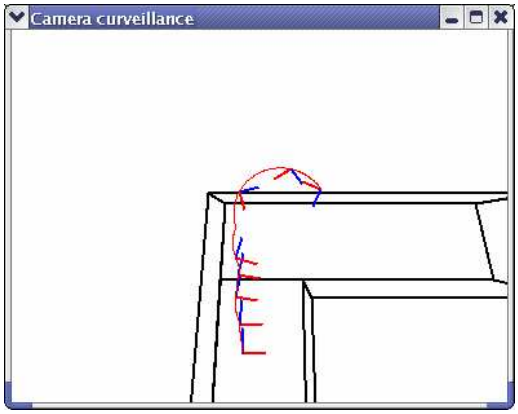


FIG. 4.33: Cas 5 (3 entrées de commande) : trajectoire réalisée par le système robotique

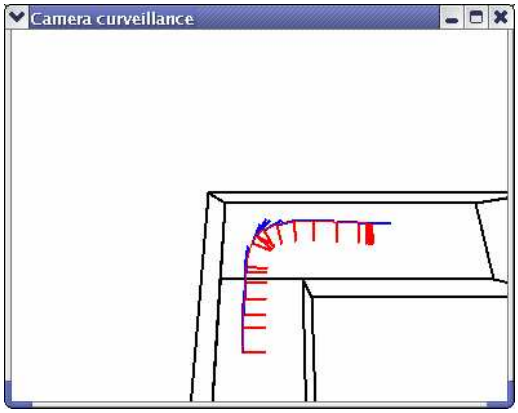


FIG. 4.34: Cas 6 (3 entrées de commande) : trajectoire réalisée par le système robotique

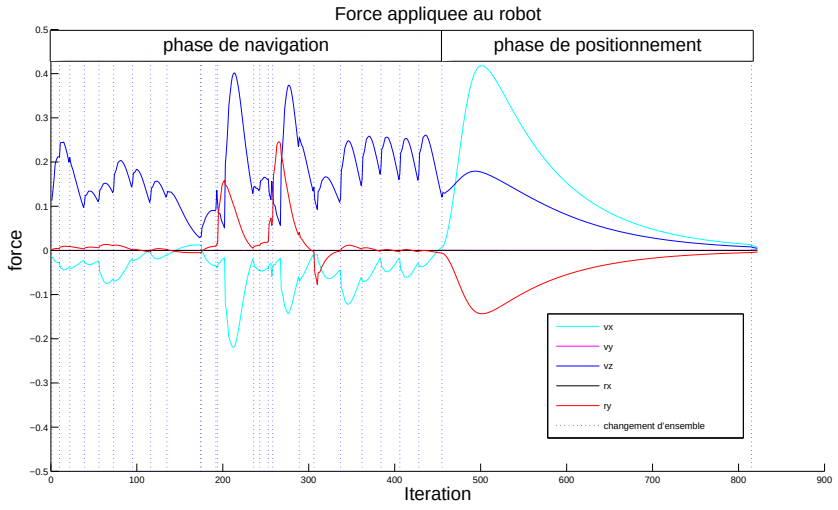


FIG. 4.35: Cas 6 : Force attractive appliquée au système robotique

4.4 Conclusion

Dans ce chapitre, nous avons proposé une méthode de contrôle des mouvements d'un système robotique pour la navigation. Dans cette méthode, il n'est pas nécessaire de connaître le modèle de la scène. La connaissance de l'environnement à traverser est donnée par un chemin d'images. Les différentes positions intermédiaires associées à ces prises de vue ne constituent pas pour autant des positions que doit successivement atteindre le système robotique. Leur fonction est d'informer le système robotique sur les amers visuels qui sont susceptibles de rentrer dans le champ de vision de la caméra durant ses mouvements.

Cette méthode ne repose pas sur une phase de planification hors-ligne. Les mouvements du système robotique sont calculés pour chaque image fournie par la caméra. Ces mouvements ont pour but de faire évoluer les amers se projetant dans le champ de vision de la caméra. Le système robotique se déplace afin de rendre observables des amers du chemin d'images qui sont proches du champ de vision de la caméra.

Pour ce faire, nous avons défini une loi de commande s'inspirant du formalisme des champs de potentiel. Différentes fonctions de potentiels ont été proposées. Celles-ci ont pour but de pénaliser les positions de l'espace où l'observabilité d'un ensemble de primitives d'intérêt n'est pas assurée de manière satisfaisante.

Dans notre approche, la détermination de la force à appliquer au système robotique ne s'effectue pas en sommant les forces induites par chacun de ces potentiels. Nous avons montré que chacun des gradients de potentiels peut être considéré comme une mesure d'erreur qui doit être régulée à zéro, opération qui est réalisée en utilisant le formalisme de l'asservissement visuel.

Nous avons présenté différents résultats expérimentaux démontrant la validité théorique de notre formalisme. Nous avons expérimenté avec succès notre approche sur un robot cartésien dans le cas d'un environnement plan.

Des simulations montrent que ce formalisme peut être utilisé dans le cas d'une scène tridimensionnelle et pour des navigations mettant en jeu l'ensemble des degrés de liberté du système robotique. Une permière perspective à ces travaux serait donc de les appliquer dans une scène de navigation $3D$, sur un système robotique réel.

Nous avons de plus réalisé des simulations d'utilisation de notre formalisme pour contrôler des systèmes robotiques se déplaçant sur un plan. Il a été montré que, sous certaines contraintes, notre approche permet d'effectuer correctement ce contrôle.

Cependant, les résultats obtenus dans le cas d'un système robotique à trois degrés de liberté montrent que notre formalisme, dans son état actuel, ne permet pas de considérer de manière satisfaisante et non supervisée tout type d'environnement.

Dans le formalisme tel que nous l'avons présenté, la notion d'obstacle n'est pas définie. La seule contrainte imposée au système robotique est de toujours conserver suffisamment d'amers dans son champ de vision. Dans les simulations réalisées sur un robot se déplaçant sur un plan, le système robotique n'a pas « conscience » des murs du couloirs. Rien ne l'empêche donc de les traverser s'il pense ainsi rendre meilleure l'observabilité des amers d'intérêt.

Une autre perspective serait donc de rendre ce formalisme applicable à des environnements de navigation restreints. La méthode développée est basée sur les champs de potentiel. Il serait donc intéressant de définir dans ce cadre un champ de potentiel répulsif afin de maintenir le système robotique à une distance respectable des obstacles que sont par exemple les murs d'un couloir, ou les bords d'une route.

Dans les différents résultats présentés (et plus particulièrement dans le cas d'un robot se déplaçant sur le plan), il peut être observé une variation de la vitesse de la caméra lors du changement d'ensemble d'intérêt. Cette variation vient de la grande variation des gradients de potentiel lors du changement. Une solution serait par exemple d'employer des techniques de gain adaptatif, afin d'éviter les trop grandes variations de forces appliquées au système robotique lors des changements d'ensemble d'intérêt.

Le fait de spécifier une décroissance exponentielle de l'erreur lors de la définition de la loi de commande est sans doute l'un des facteurs causant cette variation des vitesses du système robotique. Il serait donc intéressant de s'intéresser à d'autres types de décroissance de l'erreur. Notamment, il pourrait être intéressant de pouvoir obtenir une vitesse (en terme de norme) qui ne soit plus dépendante des positions ou images intermédiaires, mais plutôt de la difficulté du mouvement demandé.

Enfin, il serait intéressant d'adapter ce formalisme afin de pouvoir prendre en compte les contraintes de non-holonomie qui caractérisent entre autres les véhicules urbains. De nombreux travaux ont mis en place des schémas de contrôle basés vision prenant en compte ces contraintes [SwainOropeza 99, Artus 03, Blanc 04]. Dans notre cas, il serait intéressant d'étudier comment nos fonctions de potentiel pourraient être adaptées, afin par exemple de fournir des forces qui seraient définies directement dans l'espace des trajectoires réalisables du système robotique non-holonome.

Conclusion et perspectives

La réalisation d'une tâche de navigation par un système robotique autonome nécessite la résolution d'un ensemble de problèmes sous-jacents. Le robot doit tout d'abord se localiser dans son environnement, en faisant appel à ses capteurs. Il doit ensuite se définir un plan de route afin de rejoindre la position désirée. Le système robotique peut alors se consacrer à la réalisation de cette tâche de navigation, guidé par son plan de route.

Ces différentes problématiques nécessitent la mise en place d'une représentation de l'environnement de navigation. Cette mémoire permet au système robotique de se localiser, en confrontant les informations qu'elle contient avec la description locale fournie par les capteurs. De même, elle donne la possibilité d'établir un lien entre la position qu'il doit atteindre et sa position courante. Enfin, cette mémoire lui fournit les informations qui lui seront nécessaires pour qu'il puisse contrôler ses mouvements jusqu'à son objectif.

La navigation robotique autonome est un problème global et complexe, qui peut se décomposer en différentes tâches comme la localisation, la planification et le contrôle des mouvements. Si ces différents thèmes ont largement étaient traités ces dernières années, les résultats obtenus ne concernent souvent qu'une seule de ces tâches. Il est alors difficile de faire le lien entre ces différentes briques, chacune s'appuyant sur des contraintes et hypothèses différentes. Pourtant, la résolution du problème global de navigation ne peut être totalement effectuée sans assurer un enchaînement cohérent de ces différentes opérations.

Les travaux présentés dans ce manuscrit proposent un formalisme homogène permettant de répondre à chacune des problématiques dérivant d'une tâche de navigation. Notre approche est centrée autour de la vision. Dans l'ensemble des opérations réalisées, les informations utilisées sont directement extraites des images acquises par une caméra embarquée.

Contributions

Un nouveau type de représentation d'environnement de navigation a été proposé. La construction de cette mémoire s'effectue automatiquement, à partir d'une base d'images décrivant la scène. Le graphe topologique que nous employons est généré à partir des amers en correspondance entre les différentes prises de vue de la base. Ce graphe se construit autour d'une notion de facilité de déplacement du système robotique. Cette propriété permet de sélectionner parmi les différents chemins possibles celui qui est le plus facilement réalisable par le robot. Deux valuations ont été proposées. La première est satisfaisante si l'on considère un robot pouvant se déplacer aisément sur tous ses degrés de liberté. La deuxième permet de prendre en considération les libertés de

mouvement des robots non-holonomes, en pénalisant les mouvements latéraux et les mouvements de rotations.

Afin de prendre en compte la grande dimension des environnements de navigation, nous avons établi un formalisme de graphes hiérarchiques. Ces derniers regroupent dans une même entité les zones entre lesquelles les mouvements du robot sont aisés. Une recherche de chemins sur ces graphes permet à chaque étape de ne considérer que les zones de l'espace qui sont susceptibles de décrire l'environnement que doit traverser le système robotique.

Pour réaliser la localisation du robot avant le début de ses déplacements, nous avons utilisé une technique issue du domaine de la recherche d'images. Nous avons montré que l'emploi des descripteurs locaux que sont les invariants photométriques permet de réaliser des localisations qualitatives très satisfaisantes. Les résultats expérimentaux démontrent que ce type de descripteur est suffisamment discriminant pour repérer le système robotique dans de grandes bases d'images décrivant des environnements urbains.

La localisation du robot durant la navigation a aussi été traitée de manière topologique : la pose du système robotique dans l'environnement n'est pas recherchée. La position relative de la caméra par rapport aux différentes images du chemin d'images suffit pour permettre la mise à jour des amers visuels. Nous avons montré que des principes de transfert d'images pouvaient être appliqués de manière automatique et générique pour localiser de nouveaux amers.

La méthode proposée n'effectue pas la recherche des nouveaux amers sur l'image entière. La reprojection des amers du chemin sur le plan image courant, au moyen de la matrice d'homographie et des termes de parallaxe, permet d'estimer la position de nouveaux amers. Il est donc possible de limiter leur recherche dans le voisinage de ces estimations. Les résultats expérimentaux montrent que la détection de l'apparition de nouveaux amers du chemin peut s'effectuer sans connaître le modèle de la scène, sans connaître *a priori* les trajectoires images des amers, et sans imposer au robot de converger vers chacune des positions intermédiaires.

Nous avons ensuite proposé une nouvelle méthode de navigation basée vision. Celle-ci n'utilise pas de modèle de la scène, mais les amers appariés sur le chemin d'images. Les mouvements du robots sont calculés en ligne, pour chaque image acquise par la caméra. Cette méthode s'inspire du formalisme des champs de potentiel. Nous avons défini de nouvelles fonctions de potentiel, dont le but est de pénaliser les configurations du système robotique n'assurant pas une bonne observabilité d'un ensemble d'amers visuels d'intérêt. L'utilisation des principes de l'asservissement visuel permet de combiner ces différents potentiels afin de déterminer les mouvements permettant de satisfaire au mieux et simultanément les différentes contraintes posées. Le changement de l'ensemble d'intérêt durant le déplacement permet de proche en proche de conduire le système robotique vers sa position désirée.

Perspectives

Il reste cependant encore beaucoup de travaux de recherche à réaliser avant de pouvoir réaliser un système de navigation totalement autonome évoluant dans un environnement réel.

De manière générale, il serait intéressant de prendre en compte les contraintes de mouvement des véhicules dans l'ensemble des processus présentés dans ce manuscrit. Cette information per-

mettrait par exemple de rendre plus robustes les différents traitements effectués sur les images, comme la mise en correspondance ou l'estimation de la géométrie épipolaire. Ces traitements pourraient de plus profiter de la connaissance *a priori* de la structure générique des environnements urbains, à savoir un plan horizontal et deux plans verticaux.

La méthode de navigation a été développée dans le cas général d'une caméra à six degrés de liberté. L'analyse de l'évolution des trajectoires de primitives lors du déplacement d'une caméra sur un plan pourrait permettre de définir des fonctions de potentiel dédiées et plus adaptées à ce type de système. Une autre perspective évidente consisterait à intégrer les contraintes de non-holonomie, afin de pouvoir adapter le schéma de commande présenté aux robots mobiles non-holonomes.

Comme nous l'avons déjà mentionné, notre schéma de navigation ne contient pas la notion d'obstacle. Il serait donc intéressant d'étendre ce formalisme en rajoutant cette contrainte. Différentes solutions sont envisageables. De la même manière que nous avons spécifié des intervalles de confiance pour la répartition des primitives sur le plan image et pour l'orientation relative de la caméra, un intervalle de confiance pourrait être défini sur les autres degrés de liberté du système robotique, afin de forcer le système robotique à ne pas trop s'éloigner de la zone définie par le chemin d'images.

Une autre solution consisterait à détecter par un capteur (vision ou laser) les délimitations de l'espace libre de navigation du robot. Une force additionnelle pourrait alors être utilisée pour contraindre le système robotique à rester à l'intérieur de cet espace.

À l'échelle d'un environnement réel, la précision de la localisation devient un problème crucial. Même si la recherche par le contenu permet d'avoir des résultats somme toute impressionnants, on peut se demander si elle sera aussi performante face à une base suffisamment complète pour décrire toute une ville. Contrairement à la communauté de recherche d'images, nous pouvons dans notre contexte espérer résoudre ce problème par l'utilisation d'informations venant d'autres capteurs. L'utilisation par exemple de capteurs GPS permettrait d'avoir une première localisation approximative du système robotique. La définition de bases locales d'images (limitées à des espaces de navigation plus restreints, comme un quartier) permettrait alors d'effectuer une localisation satisfaisante, sur une base d'images de taille beaucoup moins importante. Éventuellement, le simple fait que le système robotique connaisse déjà cette information symbolique, ou qu'un opérateur la lui donne, permettrait de sélectionner la base d'images par rapport à laquelle le système robotique doit se repérer.

Du point de vue de la recherche de chemins, la dimension de l'espace à couvrir pose aussi un problème non trivial. Nous avons vu que l'utilisation de graphes hiérarchiques permet de capturer implicitement la topologie d'un environnement de navigation, et éventuellement de décomposer la phase de recherche en un ensemble de déplacements entre différentes zones de l'espace. Mais il reste cependant à prouver que cette méthode permet efficacement, et surtout dans des temps raisonnables, de traiter des problèmes à l'échelle de la navigation dans une ville. L'utilisation de Système d'Information Géographique (SIG) semble dans ce contexte une solution prometteuse. Ces systèmes permettraient en effet d'associer les différentes images de la base à des entités physiques, comme des routes. La recherche de chemin serait confiée tout d'abord au SIG, permettant de définir un ensemble de routes à traverser pour atteindre la position désirée. Cette recherche haut niveau permettrait de limiter considérablement l'espace de recherche d'un chemin d'images, en la confinant sur des zones topologiques.

Une autre perspective intéressante concerne la coopération multi-capteurs pour la tâche de navigation. La fusion des informations fournies par les différents capteurs peut en effet rendre plus précise la localisation du système durant la navigation. De plus, il serait aussi possible de donner une confiance aux données des différents capteurs en fonction de leur capacité à décrire l'environnement local traversé (une caméra serait par exemple peu efficace dans un environnement très dégagé).

Un autre challenge posé par une application grandeur nature vient de la propriété dynamique de l'environnement. Du point de vue de la représentation de l'environnement, un enjeu de taille est de tâcher de conserver un modèle cohérent malgré les évolutions de la scène. Une perspective intéressante serait donc de rajouter au système la possibilité de contrôler la validité de sa représentation, et éventuellement de modifier si nécessaire sa base d'information au cours de la réalisation d'une tâche de navigation.

Annexe A

Estimation des homographies et des parallaxes à partir de trois prises de vue

Nous recherchons un ensemble de trois points, communs aux trois images ψ_{i-1}, ψ_i et ψ_{i+1}, définissant le plan de référence permettant de déterminer la matrice de collinéation. Cette recherche s'effectue sans a priori sur la structure de la scène observée.

La recherche s'effectue par une méthode de sélection inspirée de la technique utilisée dans [Zhang 94]. Un ensemble de triplet est tiré de manière aléatoire, et celui minimisant un critère fixé est ainsi choisi comme définissant le plan de référence.

Pour que le plan soit stable plusieurs contraintes doivent être respectées par ces trois points. La vérification de ces contraintes permet d'éliminer les triplets non adaptés avant d'estimer les homographies.

Les points formant le plan doivent être communs aux trois images. Nous limitons donc la recherche de la base sur ces points-là, à savoir $\mathcal{M}_B = \mathcal{M}_{i-1} \bigcap \mathcal{M}_i$.

Pour effectuer le tirage aléatoire, nous définissons un espace dont les dimensions correspondent aux coordonnées minimales et maximales des primitives de $\mathcal{M}_B$ dans la vue ψ_i. Cet espace est partagé en $b \times b$ *buckets* (comme sur la figure A.1).

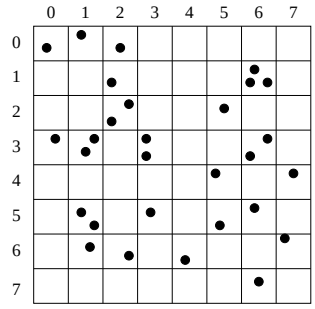


FIG. A.1: Illustration de la technique du « bucket »

En fonction de leurs coordonnées, tous les points de $\mathcal{M}_B$ sont associés à un bucket. Les buckets ne contenant pas de points sont ignorés dans la suite. Pour générer les ensembles de trois points, trois buckets sont tout d'abord choisis aléatoirement. Un point dans chacun de ces trois buckets est ensuite sélectionné.

Nous rajoutons de plus deux contraintes lors de la validation d'un triplet de point $\mathbf{x}_{p_1}$, $\mathbf{x}_{p_2}$ et $\mathbf{x}_{p_3}$:

– l'angle formé par les droites $(\mathbf{x}_{p_1}\mathbf{x}_{p_2})$ $(\mathbf{x}_{p_2}\mathbf{x}_{p_3})$ doit être suffisamment grand :

$$\frac{\|\overrightarrow{\mathbf{x}_1\mathbf{x}_2} \wedge \overrightarrow{\mathbf{x}_3\mathbf{x}_2}\|}{\|\overrightarrow{\mathbf{x}_1\mathbf{x}_2}\|\|\overrightarrow{\mathbf{x}_3\mathbf{x}_2}\|} > \epsilon$$

– les trois primitives ne doivent pas être proches du bord de l'image.

Le nombre de tirage est clairement dépendant du nombre de primitives en correspondance. En pratique, une centaine de tirages suffit pour obtenir une base satisfaisante. Pour chaque triplet de points candidats, nous calculons les matrices d'homographie ${}^1\mathbf{H}_{p_2}$ et ${}^3\mathbf{H}_{p_2}$ respectivement entre les vues ψ_2-ψ_1, et ψ_2-ψ_3. La décomposition de ces deux matrices nous fournit des ensembles de mouvements possibles, $\left({}^1\mathbf{R}_2, {}^1\mathbf{t}_{d_2}, \mathbf{n}_2\right)$ et $\left({}^3\mathbf{R}_2, {}^3\mathbf{t}_{d_2}, \mathbf{n}_2\right)$. La vue et le plan de référence étant identiques pour les deux homographies, les deux décompositions doivent nous fournir la même normale. Nous choisissons donc les deux décompositions qui sont telles que les deux normales associées forment l'angle de valeur minimal.

Nous pouvons alors déterminer la projection du centre optique de la deuxième caméra dans les vues ψ_1 et ψ_3 ;

$$\mathbf{c_1} = \mathbf{K}^2\mathbf{t}_{d_1}, \quad \mathbf{c_3} = \mathbf{K}^3\mathbf{t}_{d_1}$$

Les termes de parallaxe peuvent alors être déterminés (voir l'équation (1.12 page 18) :

$${}^i\beta_j = -\frac{\left({}^i\mathbf{H}_{p_2}{}^2\mathbf{x}_p \wedge {}^i\mathbf{x}_{p,j}\right)^\top \left(\mathbf{c}_i \wedge {}^i\mathbf{x}_{p_2}\right)}{\|\mathbf{c}_i \wedge {}^i\mathbf{x}_{p_2}\|^2},$$

pour $i = 1, 2$. Une mesure de qualité est obtenue en comparant l'erreur de reprojection réalisée par l'homographie avec la position réelle du point dans l'image :

$$\varepsilon = \sum_i \|{}^i\mathbf{x}_{p_j} - {}^i\widehat{\mathbf{x}}_{p_j}\|,$$

avec

$${}^i\widehat{\mathbf{x}}_{p_j} \propto {}^i\mathbf{H}_{p_2}{}^2\mathbf{x}_{p_j} + {}^i\beta_j\mathbf{c}_i,$$

Nous choisissons alors le plan de référence fournissant le résidu ϵ le plus faible pour les deux ensembles $\mathcal{M}_1$ et $\mathcal{M}_2$. Il faut noter que ce formalisme permet par la même occasion d'éliminer les correspondances qui ne sont pas correctement exprimées par la reprojection. Pour le meilleur plan, nous rejetons les couples de points dont l'erreur de reprojection est supérieure à dix pixels.

Bibliographie

[Amsaleg 01] L. Amsaleg, P. Gros. – Content-based retrieval using local descriptors : problems and issues from a database perspective. *Pattern Analysis and Applications*, 4(2/3) :108–124, 2001.

[Andreff 00] N. Andreff, B. Espiau, R. Horaud. – Visual servoing from lines. – IEEE *International Conference on Robotics and Automation*, pp. 2070–2075, San Francisco, États-Unis, avril 2000.

[Argyros 01] A. Argyros, C. Bekris, S. Orphanoudakis. – Robot homing based on corner tracking in a sequence of panoramic views. – IEEE *Conference on Computer Vision and Pattern Recognition*, Hawaii, États-Unis, décembre 2001.

[Arikan 01] O. Arikan, S. Chenney, D. A. Forsyth. – Efficient multi-agent path planning. – Springer-Verlag (édité par), *Eurographic workshop on Computer animation and simulation*, pp. 151–152, New-York, États-Unis, 2001.

[Arnaud 04] E. Arnaud, E. Mémin, B. Cernuschi-Frias. – Conditional filters for image sequence based tracking - application to point tracking. IEEE *Transactions on Image Processing*, 2004.

[Artus 03] G. Artus, P. Morin, C. Samson. – Tracking of an omnidirectional target with a nonholonomic mobile robot. – *International Conference on Advanced Robotics*, Coimbra, Portugal, juillet 2003.

[Avidan 97] S. Avidan, A. Shashua. – Novel view synthesis in tensor space. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 1034–1043, Puerto Rico, juin 1997.

[Avnaim 88] F. Avnaim, J.D. Boissonat, B. Faverjon. – *A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles*. – Rapport de Recherche n890, Sophia-Antipolis, INRIA, 1988.

[Baker 99] S. K. Baker, S. Nayar. – A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision*, 2(35) :1–22, novembre 1999.

[Baker 04] S. Baker, I. Matthiews. – Lucas-kanade 20 years on : a unifying approach. *International Journal of Computer Vision*, 56(3) :221–255, 2004.

[Barraquand 90] J. Barraquand, J. Latombe. – A monte carlo algorithm for path planning with many degrees of freedom. – IEEE *International Conference on Robotics and Automation*, pp. 1712–1717, 1990.

[Barraquand 92] J. Barraquand, B. Langlas, J.C. Latombe. – Numerical potential field techniques for robot path planning. IEEE *Transactions on System Science and Cybernetics*, 22 :224–241, 1992.

[Benhimane 04] S. Benhimane, E. Malis. – Real-time image-based tracking of planes using efficient second-order minimization. – IEEE *International Conference on Intelligent Robots and Systems*, octobre 2004.

[Bernardino 99] J. Bernardino, A. Santos-Victor. – Binoclar visual tracking : integration of perception and control. IEEE *Transactions on Robotics and Automation*, 6(15), 1999.

[Berrani 03] S.A. Berrani, L. Amsaleg, P. Gros. – Approximate searches : k-neighbors+precision. – *ACM International Conference on Information and Knowledge Management*, Louisiane, États-Unis, 2003.

[Berrani 04] S.A. Berrani. – *Recherche approximative de plus proches voisins avec contrôle probabiliste de la précision ; application à la recherche d'images par le contenu*. – Thèse de Doctorat, Université de Rennes 1, février 2004.

[Binford 71] T.O. Binford. – Visual perception by computer. – IEEE *Conference on Systems Science and Cybernetics*, Miami, États-Unis, 1971.

[Blanc 04] G. Blanc, O. Ait-Ader, Y. Mezouar, T. Chateau, P. Martinet. – Autonomous image-based navigation in indoor environment. – *Symposium on Intelligent Autonomous Vehicles*, Lisbonne, Portugal, juillet 2004.

[Borenstein 96] J. Borenstein, B. Everett, L. Feng. – *Navigating mobile robots : systems and techniques*. – A. K. Peters, Ltd, Wellesley, 1996.

[Boufama 95] B. Boufama, R. Mohr. – Epipole and fundamental matrix estimation using the virtual parallaxe property. – IEEE *International Conference on Computer Vision*, pp. 1030–1038, Cambridge, États-Unis, juin 1995.

[Brooks 83] R.A. Brooks. – Solving the find-path problem by representing free space as generalized cones. IEEE *Transactions on Systems, Man and Cybernetics*, 13(3) :190–197, 1983.

[Bruce 02] J. Bruce, M. Veloso. – Real-time randomized path planning for robot navigation. – IEEE *International Conference on Intelligent Robots and Systems*, 2002.

[Burgard 98] W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, S. Thrun. – The interactive museum tour-guide robot. – *Conference on Artificial intelligence*, pp. 11–18, Madison, États-Unis, juillet 1998.

[Burschka 01] D. Burschka, G. D. Hager. – Vision-based control of mobile robots. – IEEE *International Conference on Robotics and Automation*, pp. 1707–1713, Séoul, Corée du Sud, mai 2001.

[Canny 88] J. T. Canny. – *The complexity of robot motion planning*. – Thèse de Doctorat, MIT Cambridge, 1988.

[Chambon 02] S. Chambon, A. Crouzil. – *Évaluation et comparaison de mesures de corrélation robustes*. – Rapport de recherche, Institut de Recherche en Informatique de Toulouse, 2002.

[Chaumette 90] F. Chaumette. – *La relation vision-commande : théorie et application à des tâches robotiques*. – Thèse de doctorat, Université de Rennes I, 1990.

[Chaumette 02] F. Chaumette. – Asservissement visuel. *La commande des robots manipulateurs*, éd. par W. Khalil, chap. 3, pp. 105–150. – Hermès, 2002.

[Chaumette 04] F. Chaumette. – Image moments : a general and useful set of features for visual servoing. IEEE *Transactions on Robotics and Automation*, 20(4) :713–723, août 2004.

[Chazelle 87] B. Chazelle. – Approximation and decomposition of shapes. *Advances in Robotics 1, (J.T. Schwartz and C.K. Yap, ed)*, pp. 145–185, 1987.

[Chesi 00] G. Chesi, E. Malis, R. Cipolla. – Automatic segmentation and matching of planar contours for visual servoing. – IEEE *International Conference on Robotics and Automation*, vol. 3, pp. 2753–2758, San Francisco, États-Unis, avril 2000.

[Chesi 03] G. Chesi, K. Hashimoto, D. Prattichizzo, A. Visino. – A switching control law for keeping features in the field of view in eye-in-eand eisual servoing. – IEEE *International Conference on Robotics and Automation*, Taipei, Taiwan, septembre 2003.

[Cobzas 01] D. Cobzas, H. Zhang. – Mobile robot localization using planar patches and a stereo panoramic model. – *Vision Interface*, pp. 94–99, Ottawa, Canada, juin 2001.

[Cobzas 03] D. Cobzas, H. Zhang, M. Jagersand. – Image-based localization with depth-enhanced image map. – IEEE *International Conference on Robotics and Automation*, Taiwan, mai 2003.

[Comport 03] A.I. Comport, M. Pressigout, E. Marchand, F. Chaumette. – A visual servoing control law that is robust to image outliers. – IEEE *International Conference on Intelligent Robots and Systems*, vol. 1, pp. 492–497, Las Vegas, États-Unis, octobre 2003.

[Cordesses 00] L. Cordesses, B. Thuilot, P. Martinet, C. Cariou. – Curved path following of a farm tractor using a cp-dgps. – *Proceedings of the 6th SYmposium on RObot COntrol*, pp. 489–494, Vienne, Autriche, septembre 2000.

[Corke 00a] P. I. Corke, S. Hutchinson. – A new partitioned approach to image-based visual servoing. – IEEE *International Conference on Decision and Control*, pp. 2521–2526, Sydney, Austalie, décembre 2000.

[Corke 00b] P. I. Corke, J. Trevelyan (édité par). – *Explicit incorporation of* $2D$ *constraints in vision based control of robot manipulators*. – Springer Verlag, mars 2000.

[Couapel 95] B. Couapel, K. Bainian. – Stereo vision with the use of a virtual plane in the space. *Chinese Journal of Electronics*, 4(2) :32–39, avril 1995.

[Cowan 99] N. J. Cowan, D. E. Koditschek. – Planar image based visual servoing as a navigation problem. – IEEE *International Conference on Robotics and Automation*, pp. 611–617, Détroit, États-Unis, mai 1999.

[Crétual 98] A. Crétual, F. Chaumette. – Image-based visual servoing by integration of dynamic measurements. – IEEE *International Conference on Robotics and Automation*, pp. 1994–2001, Louvain, Belgique, mai 1998.

[Cui 03] Y. Cui, S. G. Shuzhi. – Autonomous vehicle positioning with gps in urban canyon environment. IEEE *Transactions on Robotics and Automation*, 19(1) :15–25, 2003.

[Curless 96] B. Curless, M. Levoy. – A volumetric method for building complex models from range images. *Computer Graphics*, 30 :303–312, 1996.

[Dao 03] N. X. Dao, B. J. You, S. R. Oh, M. Hwangbo. – Visual self-localization for indoor mobile robots using natural lines. – IEEE *International Conference on Intelligent Robots and Systems*, pp. 1252–1255, 2003.

[Daucher 97] N. Daucher, M. Dhome, J. T. Lapresté, G. Rives. – Speed command of a robotic system ny monocular pose estimate. – IEEE *International Conference on Intelligent Robots and Systems*, vol. 1, pp. 55–62, Grenoble, France, septembre 1997.

[Davison 98] A. J. Davison, A. W. Murray. – Mobile robot localisation using active vision. – *European Conference on Computer Vision*, pp. 809–825, Freiburg, Allemagne, 1998.

[Davison 03] A.J. Davison. – Real-time simultaneous localisation and mapping with a single camera. – IEEE *International Conference on Computer Vision*, Nice, France, octobre 2003.

[DeLaTorre 01] F. De La Torre, M. J. Black. – Robust principal component analysis for computer vision. – IEEE *International Conference on Computer Vision*, vol. 1, pp. 362–369, Vancouver, Canada, juillet 2001.

[Dellaert 99] F. Dellaert, W. Burgard, D. Fox, S. Thrun. – Using the condensation algorithm for robust, vision-based mobile robot localization. IEEE *Conference on Computer Vision and Pattern Recognition*, juin 1999.

[Dementhon 95] D. Dementhon, L.S. Davis. – Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15 :123–141, 1995.

[Deriche 94] R. Deriche, Z. Zhang, Q.T. Luong, O.D. Faugeras. – Robust recovery of the epipolar geometry for an uncalibrated stereo rig. – *European Conference on Computer Vision*, pp. 567–576, 1994.

[DeSouza 02] G. N. DeSouza, A. C. Kak. – Vision for mobile robot navigation : a survey. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 24(2), février 2002.

[Devernay 95] F. Devernay, O.D. Faugeras. – Automatic calibration and removal of distortion from scenes of structured environments. *SPIE*, 2576, Juillet 1995.

[ElNajjar 03] M. E. El Najjar, V. Cherfaoui, P. Bonnifait, C. Royere. – Élaboration de fonctions de croyance à partir de données GPS et SIG : étude de stratégies de fusion pour la localisation d'un véhicule. *TSI. Technique et science informatiques"*, pp. 935 – 964. – Hermès, 2003.

[Espiau 02] F.X. Espiau, E. Malis, P. Rives. – Robust features tracking for robotic applications : Towards $2\frac{1}{2}d$ visual servoing with natural images. – IEEE *International Conference on Robotics and Automation*, pp. 574–579, Washington, États-Unis, mai 2002.

[Faugeras 88] O.D. Faugeras, F. Lustman. – Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2 :485–508, 1988.

[Faugeras 92] O.D. Faugeras, P. Fua, B. Hotz, R. Ma, L. Robert, M. Thonnat, Z. Zhang. – Quantitative and qualitative comparisons of some area and feature-based stereo algorithm. *Robust Computer Vision*, pp. 1–26, 1992.

[Faugeras 93] O. Faugeras. – *Three-dimensional computer vision : a geometric viewpoint*. – MIT Press, Cambridge, Massachusetts, 1993.

[Florack 94] L. Florack, B. ter Haar Romeny, J. Koenderink, M. Viergever. – General intensity transformation and differential invariants. *Journal of Mathematical Imaging and Vision*, 4(2) :171–187, 1994.

[Franz 00] M. Franz, H. Mallot. – Biomimetic robot navigation. *Robotics and Autonomous Systems*, pp. 133–153, 2000.

[Froidevaux 92] C. Froidevaux, M. Gaudel, M. Soria. – *Types de données et algorithmes*. – Mc Graw-Hill, 1992.

[Garcia 04] N. Garcia, E. Malis. – Preserving the continuity of visual servoing despite changing image features. – IEEE *International Conference on Intelligent Robots and Systems*, Sendai, Japon, octobre 2004.

[Gaspar 00] J. Gaspar, J. Santos-Victor. – Vision-based navigation and environmental representations with an omni-directionnal camera. IEEE *Transactions on Robotics and Automation*, 16(6) :890–898, 2000.

[Ghosh 87] S. K. Ghosh, D. M. Mount. – An output sensitive algorithm for computing visibility graphs. – IEEE *Symposium on Foundations of Computer Science*, Los Angeles, États-Unis, 1987.

[Giachetti 00] A. Giachetti. – Matching techniques to compute image motion. *Image And Vision Computing*, 18(3) :247–260, 2000.

[GonzalesBarbosa 02] J.-J. Gonzales-Barbosa, S. Lacroix. – Rover localization in natural environments by indexing panoramic images. – IEEE *International Conference on Robotics and Automation*, pp. 1365–1370, Washington, États-Unis, mai 2002.

[Gouiffès 04] M. Gouiffès, C. Collewet, C. Maloigne-Fernandez, A. Trémeau. – Suivi de points : extension aux objets non-lambertiens. – *Journées des Jeunes Chercheurs en Robotique*, 2004.

[Gracias 03] N. Gracias, S. Zwaan, A. Bernardino, J. Santos-Victor. – Mosaic-based navigation for autonomous underwater vehicles. IEEE, *Journal of Oceanic Engineeing*, 28(4), octobre 2003.

[Gros 98] P. Gros, O. Bournez, E. Boyer. – Using local planar geometric invariants to match and model images of line segments. *Computer Vision and Image Understanding*, 69(2) :135–155, 1998.

[Guivant 01] J.E. Guivant, E.M. Nebot. – Optimization of the simultaneous localization and map-building algorithm for real-time implementation. IEEE *Transactions on Robotics and Automation*, 17(3) :242–257, juin 2001.

[Hager 97a] G. D. Hager, D. Kriegman, E. Yeh, C. Rasmussen. – Image-based prediction of landmark features for mobile robot navigation. – IEEE *International Conference on Robotics and Automation*, pp. 1040–1046, 1997.

[Hager 97b] G.D. Hager. – A modular system for robust positioning using feedback from stereo vision. IEEE *Transactions on Robotics and Automation*, 13(4), août 1997.

[Hager 98a] G. D. Hager, P. Belhumeur. – Efficient region tracking with parametric models of geometry and illumination. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 20(10) :1025–1039, 1998.

[Hager 98b] G. D. Hager, D. J. Kriegman, A. S. Georghiades, O. Ben-Shalar. – Toward domain-independent navigation : dynamic vision and control. – IEEE *International Conference on Decision and Control*, pp. 1040–1046, Tampa, décembre 1998.

[Harris 88] C. Harris, M. Stephens. – A combined corner and edge detector. – *Alvey Vision Conf.*, pp. 147–151, University of Manchester, England, septembre 1988.

[Hartley 92] R. Hartley. – Estimation of relative camea positions for uncalibrated cameras. – *European Conference on Computer Vision and Pattern Recognition*, 1992.

[Hartley 93] R. Hartley. – Euclidean reconstruction from uncalibrated views. *Applications of Invariance in Computer Vision*, éd. par J. Mundy, A. Zisserman, pp. 237–256. – Springer Verlag, 1993.

[Hartley 94] R. Hartley. – Projective reconstruction from line correspondences. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 903–907, Seattle, États-Unis, 1994.

[Hartley 97a] R. Hartley. – In defense of the eight-point algorithm. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(6) :580–593, juin 1997.

[Hartley 97b] R. Hartley. – Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 19(2) :125–140, 1997.

[Hartley 00] R. Hartley, A. Zisserman. – *Multiple view geometry in computer vision*. – Cambridge University Press, Angleterre, 2000.

[Hashimoto 93] K. Hashimoto (édité par). – *Visual Servoing Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*. – World Scientific Publishing, River Edge, NJ, 1993.

[Hayet 02] J. B. Hayet, F. Lerasle, M. Devy. – A Visual Landmark Framework for Indoor Mobile Robot Navigation. – IEEE *International Conference on Robotics and Automation*, pp. 3942–3947, Washington, États-Unis, mai 2002.

[Heikkilä 00] J Heikkilä. – Geometric camera calibration using circular control points. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(10) :1066–1077, octobre 2000.

[Hendrickson 93] B. Hendrickson, R. Leland. – *A multi-level algorithm for partitioning graphs*. – Rapport de recherche, Sandia National Laboratories, 1993.

[Horaud 93] O. Horaud, R. Monga. – *Vision par ordinateur, outils fondamentaux*. – Hermès, Paris, France, 1993.

[Hosoda 95] K. Hosoda, K. Sakamoto, M. Asada. – Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without $3d$ reconstruction. – IEEE *International Conference on Intelligent Robots and Systems*, vol. 1(3), pp. 29–34, Pittsburgh, États-Unis, août 1995.

[Huang 97] J. Huang, S. R. Kumar, M. Mitra, W. J. Zhu, R. Zabih. – Image indexing using color correlograms. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 762–768, Puerto Rico, États-Unis, 1997.

[Huber 81] P.-J. Huber. – *Robust statistics*. – Wiler, New York, États-Unis, 1981.

[Hutchinson 96] S. Hutchinson, G.D. Hager, P.I. Corke. – A tutorial on visual servo control. IEEE *Transactions on Robotics and Automation*, 12(5) :651–670, octobre 1996.

[Inoue 93] H. Inoue, M. Inaba, T. Mori, T. Tachikawa. – Real-time robot vision system based on correlation technology. – *International Symposium on Industrial Robots*, pp. 675–680, 1993.

[Jin 01] H. Jin, P. Favaro, S. Soatto. – Real-time feature tracking and outlier rejection with changes in illumination. – IEEE *International Conference on Computer Vision*, vol. 1, pp. 684–689, Vancouver, Canada, juillet 2001.

[Jogan 99] M. Jogan, A. Leonardis. – Panoramic eigenimages for spatial localisation. – *Computer Analysis of Images and Patterns*, pp. 558–567, Ljubljana, Slovénie, 1999.

[Jolliffe 86] I. T. Jolliffe. – *Principal Component Analysis*. – Springer, 1986.

[Jones 97] S. Jones, C. Andersen, J. L. Crowley. – Appearance based processes for visual navigation. IEEE *International Conference on Intelligent Robots and Systems*, septembre 1997.

[Jurie 02] F. Jurie, M. Dhome. – Hyperplane approximation for template matching. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 24(7) :996–1000, juillet 2002.

[Karypis 98] G. Karypis, V. Kumar. – A fast and highly quality multilevel scheme for partitioning irregular graphs. *SIAM Journal of Scientific Computing*, 20(1) :359–392, 1998.

[Katsura 03] H. Katsura, J. Miura, M. Hild, Y. Shirai. – A view-based outdoor navigation using object recognition robust to changes of weather and seasons. – IEEE *International Conference on Intelligent Robots and Systems*, Las Vegas, États-Unis, octobre 2003.

[Kavraki 96] L.E. Kavraki, P. Svestka, J.C. Latombe, M. Overmars. – Probabilistic roadmap for path planning in high-dimensional configuration spaces. IEEE *Transactions on Robotics and Automation*, 12(4) :566–580, 1996.

[Khatib 86] O. Khatib. – Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1) :90–98, 1986.

[Koditschek 87] D. E. Koditschek. – Exact robot navigation by means of potential functions : some topological considerations. – IEEE *International Conference on Robotics and Automation*, pp. 1–6, Raleigh, 1987.

[Koren 91] Y. Koren, J. Borenstein. – Potential field methods and their inherent limitations for mobile robot navigation. – IEEE *International Conference on Robotics and Automation*, pp. 1398–1404, Sacramento, États-Unis, avril 1991.

[Košecká 03] J. Košecká, L. Zhou, P. Barber, Z. Duric. – Qualitative image based localization in indoor environments. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 3–10, Madison, États-Unis, juin 2003.

[Kröse 99] B. Kröse, R. Bunschoten, N. Vlassis, Y. Motomura. – Appearance based robot localization. – *International Joint Conference on Artificial Intelligence*, pp. 53–58, 1999.

[Lamarche 04] F. Lamarche, S. Donikian. – Crowds of virtual humans : a new approach for real time navigation in complex and structured environments. – *Computer Graphics Forum*, vol. 23(3). Eurographics, 2004.

[Latombe 93] J.C. Latombe. – *Robot Motion Planning*. – Kluwer, 1993.

[Laumond 87] J. P. Laumond. – Obstacle growing in a non-polygonal world. *Information Processing Letters*, 25(1) :41–50, 1987.

[Laumond 00] J. P. Laumond, T. Siméon. – Notes on visibility roadmaps and path planning. – *International Workshop on the Algorithmic Foundations of Robotics*, Hanover, États-Unis, mars 2000.

[Laumond 01] J.P. Laumond (édité par). – *La robotique mobile*. – Traité IC2, série systèmes automatisées, Hermes Science Publications, 2001.

[Laveau 96] S. Laveau. – *Géométrie d'un système de N caméras. Théorie, estimation, et applications*. – Thèse de Doctorat, École Polytechnique, mai 1996.

[Lingrand 99] D. Lingrand. – *Approximations de la projection perspective : modèle combiné et étude des singularités homographiques*. – Rapport de Recherche n 3682, INRIA, 1999.

[LonguetHiggins 81] H.C. Longuet-Higgins. – A computer algorithm for reconstructing a scene from two projections. *Nature*, 293 :133–135, septembre 1981.

[Lowe 01] D. G. Lowe. – Local feature view clustering for $3d$ object recognition. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 682–688, Kauai, Hawaii, décembre 2001.

[Lowe 04] D. G. Lowe. – Distinctive image features from scale-inavariant keypoints. *International Journal of Computer Vision*, 60(2) :91–110, 2004.

[LozanoPerez 79] T. Lozano-Perez, M. A. Wesley. – An algorithm for planning collision-free paths among polyhedral obstacles. *Communication of the ACM*, 22(10) :560–570, 1979.

[Lucas 81] B. Lucas, B. Kanade. – An iterative image registration technique with an application to stereo vision. – *International Joint Conference on Artificial Intelligence*, pp. 674–679, avril 1981.

[Luong 96] Q. T. Luong, O. Faugeras. – The fundamental matrix : theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1), janvier 1996.

[Malis 98] E. Malis. – *Contributions à la modélisation et à la commande en asservissement visuel*. – Thèse de doctorat, Université de Rennes I, 1998.

[Malis 99] E. Malis, F. Chaumette, S. Boudet. – 2 1/2 d visual servoing. IEEE *Transactions on Robotics and Automation*, 15(2) :238–250, avril 1999.

[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*, 37(1) :79–97, juin 2000.

[Malis 02] E. Malis, F. Chaumette. – Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods. IEEE *Transactions on Robotics and Automation*, 18(2) :176–186, avril 2002.

[Malis 04] E. Malis. – Improving vision-based control using efficient second-order minimization techniques. – IEEE *International Conference on Robotics and Automation*, New Orleans, États-Unis, avril 2004.

[Martinet 96] P. Martinet, J. Gallice, D. Khadraoui. – Vision based control law using 3d visual features. – *World Automation Congress, Robotics and Manufacturing systems*, vol. 3, pp. 497–502, Montpellier, France, mai 1996.

[Martinet 99] P. Martinet, J. Gallice. – Position based visual servoing using a nonlinear approach. – IEEE *International Conference on Intelligent Robots and Systems*, vol. 1, pp. 531–536, Kyongju, Corée du Sud, octobre 1999.

[Matsumoto 96] Y. Matsumoto, M. Inaba, H. Inoue. – Visual navigation using view-sequenced route representation. – IEEE *International Conference on Robotics and Automation*, pp. 83–88, Minneapolis, États-Unis, 1996.

[Matsumoto 00a] Y. Matsumoto, M. Inaba, H. Inoue. – Exploration and navigation in corridor environment based on omni-view sequence. – IEEE *International Conference on Intelligent Robots and Systems*, pp. 1505–1510, Takamatsu, Japon, octobre 2000.

[Matsumoto 00b] Y. Matsumoto, M. Inaba, H. Inoue. – View-based approach to robot navigation. – IEEE *International Conference on Intelligent Robots and Systems*, pp. 1702–1708, Takamatsu, Japon, octobre 2000.

[Matthews 04] I. Matthews, T. Ishikawa, S. Baker. – The template update problem. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 26(6) :810 – 815, juin 2004.

[Menegatti 04] E. Menegatti, T. Maeda, H. Ishiguro. – Image-based memory for robot navigation using properties of omnidirectional images. *Robotics And Autonomous Systems*, 47 :251–267, 2004.

[Mezouar 01] Y. Mezouar. – *Planification de trajectoires pour l'asservissement visuel*. – Thèse de doctorat, Université de Rennes I, 2001.

[Mezouar 02a] Y. Mezouar, F. Chaumette. – Avoiding self-occlusions and preserving visibility by path planning in the image. *Robotics and Autonomous Systems*, 41(2) :77–87, novembre 2002.

[Mezouar 02b] Y. Mezouar, F. Chaumette. – Path planning for robust image-based control. IEEE *Transactions on Robotics and Automation*, 18(4) :534–549, août 2002.

[Mezouar 02c] Y. Mezouar, A. Remazeilles, P. Gros, F. Chaumette. – Image interpolation for image-based control under large displacement. – IEEE *International Conference on Robotics and Automation*, vol. 3, pp. 3787–3794, Washington, États-Unis, mai 2002.

[Mezouar 03] Y. Mezouar, F. Chaumette. – Optimal camera trajectory with image-based control. *International Journal of Robotics Research*, 22(10) :781–804, octobre 2003.

[Mezouar 04] Y. Mezouar, H. Haj Abdelkader, P. Martinet, F. Chaumette. – Central catadioptric visual servoing from 3d straight lines. – IEEE *International Conference on Intelligent Robots and Systems*, Sendai, Japon, septembre 2004.

[Mikolajczyk 01] K. Mikolajczyk, C. Schmid. – Indexing based on scale invariant interest points. – IEEE *International Conference on Computer Vision*, pp. 525–531, Vancouver, Canada, 2001.

[Mohr 96] R. Mohr, B. Triggs. – Projective geometry for image analysis. – *International Symposium of Photogrammetry and Remote Sensing*, juillet 1996.

[Nilsson 69] N.J. Nilsson. – A mobile automaton : An application of artificial intelligence. – *International Joint Conference on Artificial Intelligence*, pp. 509–521, Washington D.C., États-Unis, 1969.

[Odobez 98] J.-M. Odobez, P. Bouthemy. – Direct incremental model-based image motion segmentation for video analysis. *Signal Processing*, 6(2) :143–155, 1998.

[Ohya 98] A. Ohya, A. Kosaka, A. Kak. – Vision-based navigation by a mobile robot with obstacle avoidance using single camera vision and ultrasonic sensing. IEEE *Transactions on Robotics and Automation*, 14(6) :969–978, décembre 1998.

[Overmars 95] M. H. Overmars, P. Svetska. – A probabilistic learning approach to motion planning. – *Workshop on Algorithmic Foundations of Robotics*, San Francisco, États-Unis, 1995.

[Paletta 01] L. Paletta, S. Frintrop, J. Hertzberg. – Robust localization using context in omnidirectionnal imaging. – IEEE *International Conference on Robotics and Automation*, pp. 2072–2077, Séoul, Corée du Sud, mai 2001.

[Papanikolopoulos 73] N. Papanikolopoulos, P. K. Khodla. – Adaptive robotic visual tracking : Theory and experiments. IEEE *Transactions on Automatic Control*, 38(3) :429–445, mars 1973.

[Parra 99] C. Parra, Murrieta-Cid R., M. Devy, M. Briot. – $3d$ modelling and robot localization from visual and range data in natural scenes. *Computer Vision Systems : First International Conference*, éd. par H. Christensen, pp. 450–468. – Springer Verlag, 1999.

[Plakas 00] C. Plakas, E. Trucco. – Developing a real-time, robust, video tracker. – *IEEE OCEANS*, pp. 1345–1353, Providence, Rhode Island, septembre 2000.

[Pressigout 04] M. Pressigout, E. Marchand. – Model-free augmented reality by virtual visual servoing. – *International Conference on Pattern Recognition*, vol. 2, pp. 887–891, Cambridge, Royaume-Uni, août 2004.

[Puzicha 97] J. Puzicha, Hofmann T., J. M. Buhmann. – Non parametric similarity measures for unsupervised texture segmentation and image retrieval. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 267–272, juin 1997.

[Rasmussen 96] C. Rasmussen, G.D. Hager. – Robot navigation using image sequences. – *National Conference on Artificial Intelligence*, vol. 2, pp. 938–943, Portland, 1996.

[Remazeilles 02] A. Remazeilles, F. Chaumette, P. Gros. – Couplage entre l'indexation d'images et l'asservissement visuel pour la réalisation de grands déplacements. – *Journées des Jeunes Chercheurs en Robotique*, Strasbourg, janvier 2002.

[Rimon 92] E. Rimon, D.E. Koditschek. – Exact robot navigation using artificial potential functions. IEEE *Transactions on Robotics and Automation*, 8 :501–518, 1992.

[Royer 04] E. Royer, M. Lhuiller, M. Dhome, T. Chateau. – Towards an alternative gps sensor in dense urban environment from visual memory. – *British Machine Vision Conference*, Londres, Royaume-Uni, septembre 2004.

[Ruf 97] A. Ruf, R. Horaud. – Visual trajectories from uncalibrated images. – R. Horaud, F. Chaumette (édité par), *Workshop on New Trends in Image-Based Robot Servoing*, pp. 83–91, Grenoble, France, septembre 1997. IEEE International Conference on Robotics and Automation.

[Rémy 98] S. Rémy. – *Étalonnage d'un système de vision embarqué*. – Thèse de Doctorat, Université Blaise Pascal, 1998.

[Samson 91] C. Samson, M. Le Borgne, B. Espiau. – *Robot control : The Task function approach*. – Clarendon Press, Oxford, 1991.

[SantosVictor 97] J. Santos-Victor, G. Sandini. – Visual behaviors for docking. *Computer Vision and Image Understanding*, 67(3) :223–238, 1997.

[Schmid 96] C. Schmid. – *Appariements d'images par invariants locaux de niveaux de gris*. – Thèse de doctorat, Institut National Polytechnique de Grenoble, 1996.

[Schmid 97] C. Schmid, R. Mohr. – Local grayvalue invariants for image retrieval. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(5) :530–534, mai 1997.

[Schmid 00] C Schmid, R. Mohr, C. Bauckhage. – Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2) :151–172, 2000.

[Se 02] S. Se, D. Lowe, J. Little. – Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, 21(8) :735–758, 2002.

[Shashua 94a] A. Shashua. – Algebraic functions for recognition. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(8) :779–789, 1994.

[Shashua 94b] A. Shashua, N. Navab. – Relative affine structure : Theory and application to $3d$ reconstruction from perspective views. – *Conference on Vision and Pattern Recognition*, pp. 483–489, 1994.

[Shashua 96] A. Shashua, N. Navab. – Relative affine structure : Canonical model for $3d$ from $2d$ geometry and applications. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 18(9) :873–883, septembre 1996.

[Shi 94] J. Shi, C. Tomasi. – Good features to track. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 593–600, Seattle, juin 1994.

[Siegwart 04] R. Siegwart, R. Nourbakhsh. – *Introduction to autonomous mobile robots*. – MIT Press, 2004.

[Sim 99] R. Sim, G. Dudeck. – Learning visual landmarks for pose estimation. – IEEE *International Conference on Robotics and Automation*, Detroit, États-Unis, mai 1999.

[Simmons 95] R. Simmons, S. Koenig. – Probabilistic robot navigation in partially observable environments. – *International Joint Conference on Artificial Intelligence*, pp. 1080 – 1087, juillet 1995.

[Simon 00] G. Simon, A. W. Fitzgibbon, A. Zisserman. – Markerless tracking using planar structures in the scene. – *International Symposium on Augmented Reality*, octobre 2000.

[Simond 04] N. Simond, P. Rives. – Trajectography of an uncalibrated stereo rig in urban e nvironments. – IEEE *International Conference on Intelligent Robots and Systems*, pp. 3381–3386, Sendai, Japon, septembre 2004.

[Smeulders 00] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain. – Content-based image retrieval at the end of the early years. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 22(12) :1349–1380, 2000.

[Smith 02] L. Smith, P. Husbands. – Visual landmark navigation through large-scale environments. – *International Workshop on Biologically-Inspired Robotics*, pp. 272–279, 2002.

[Stolfi 91] J. Stolfi. – *Oriented Projective Geometry*. – Academic Press, 237p., Boston, États-Unis, 1991.

[Stricker 94] M. Stricker, M. Swain. – The capacity of color histogram indexing. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 704–708, Seattle, États-Unis, juin 1994.

[SwainOropeza 99] R. Swain Oropeza. – *Contrôle de tâches référencées vision d'un robot mobile en milieu structuré*. – Thèse de Doctorat, Institut National Polytechnique de Toulouse, 1999.

[Tahri 04] O. Tahri, F. Chaumette. – Image moments : Generic descriptors for decoupled image-based visual servo. – IEEE *International Conference on Robotics and Automation*, vol. 2, pp. 1185–1190, New Orleans, États-Unis, avril 2004.

[Thrun 96] S. Thrun, A. Bücken. – Integrating grid-based and topological maps for mobile robot navigation. – *National Conference on Artificial Intelligence*, Portland, États-Unis, août 1996.

[Thrun 00] S. Thrun, W. Burgard, D. Fox. – A real time algorithm for mobile robot mapping with applications to multi-robot and $3d$ mapping. – IEEE *International Conference on Robotics and Automation*, San Francisco, États-Unis, avril 2000.

[Tissainayagam 04] P. Tissainayagam, D. Suter. – Assessing the performance of corner detectors for point feature tracking applications. *Image and Vision Computing*, 22(8) :663–679, août 2004.

[Triggs 00a] B. Triggs. – Plane + parallax, tensors and factorization. – *European Conference on Computer Vision*, pp. 522–538, Dublin, Irelande, 2000.

[Triggs 00b] B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon. – Bundle adjustment, a modern synthesis. *Vision Algorithms : Theory and Practice*, éd. par W. Triggs, A. Zisserman, R. Szeliski, pp. 298–375. – Springer Verlag, 2000.

[Trémeau 04] A. Trémeau, C. Fernandez-Maloigne, P. Bonton. – *Image numérique couleur : de l'acquisition au traitement*. – Dunod, Paris, 2004.

[Tsai 86] R. Y. Tsai. – An efficient and accurate camera calibration technique dor $3d$ machine vision. – IEEE *Conference on Computer Vision and Pattern Recognition*, pp. 364–374, Miami Beach, Floride, 1986.

[Ulrich 00] I. Ulrich, I. Nourbakhsh. – Appearance-based place recognition for topological localization. – IEEE *International Conference on Robotics and Automation*, pp. 1023–1029, San Francisco, États-Unis, avril 2000.

[Vassallo 00] R. Vassallo, H. Schneebeli, J. Santos-Victor. – Visual servoing and appearance for navigation. *Robotics and Autonomous Systems*, 31 :87–97, avril 2000.

[Veltkamp 00] R. C. Veltkamp, M. Ranase. – *Content-based image retrieval systems : a survey*. – Rapport de recherche, Department of Computing Science, Utrecht Univesity, 2000.

[Victorino 00] A.C. Victorino, P. Rives, Borrelly. – Localization and map building using a sensor-based control strategy. – IEEE *International Conference on Intelligent Robots and Systems*, pp. 937–942, Takamatsu, Japon, octobre 2000.

[Victorino 01] A.C. Victorino, P. Rives, Borrelly. – Mobile robot navigation using a sensor-based control strategy. – IEEE *International Conference on Robotics and Automation*, pp. 2753–2758, Séoul, Corée du Sud, mai 2001.

[Victorino 04] A.C. Victorino, P. Rives. – An hybrid representation well-adapted to the exploration of large scale indoors environments. – IEEE *International Conference on Robotics and Automation*, pp. 2930–2935, New Orleans, États-Unis, avril 2004.

[Wilson 96] W.J. Wilson, C.C. Williams Hulls, G.S. Bell. – Relative end-effector control using cartesian position based visual servoing. IEEE *Transactions on Robotics and Automation*, 12(5) :684–696, 1996.

[Wolf 02] J. Wolf, W. Burgard, H. Burkhardt. – Robust vision-based localization for mobile robots using an image retrieval system based on invariant features. – IEEE *International Conference on Robotics and Automation*, Washington DC, États-Unis, mai 2002.

[Yap 85] C.K. Yap. – *An $\mathcal{O}(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments*. – Rapport de recherche, Robotics Laboratory, Courant Institute, New-York University, 1985.

[Yuen 02] D. Yuen, B. MacDonald. – Natural landmark based localisation system using panoramic images. – IEEE *International Conference on Robotics and Automation*, vol. 1, pp. 915–20, Washington DC, États-Unis, mai 2002.

[Zhang 94] Z. Zhang, R. Deriche, Q. Luong, O. Faugeras. – A robust approach to image matching : Recovery of the epipolar geometry. *International Symposium of Young Investigators on Information-Computer-Control*, 1994.

[Zhang 95a] Z. Zhang. – On the epipolar geometry between two images with lens distorsion. – *International Conference on Pattern Recognition*, vol. 1, pp. 407–411, Vienne, Autriche, août 1995.

[Zhang 95b] Z. Zhang, A. Hanson. – Scaled euclidean 3d reconstruction based on externally u ncalibrated cameras. – IEEE *Symposium on Computer Vision*, 1995.

[Zhang 98] Z. Zhang. – Determining the epipolar geometry and its uncertainty - a review. *International Journal of Computer Vision*, 27(2) :161–195, mars 1998.

[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*, pp. 1271–1276, Taipei, Taiwan, septembre 2003.

[Zhu 91] D. Zhu, J.C. Latombe. – New heuristic algorithms for efficient hierarchical path planning. IEEE *Transactions on Robotics and Automation*, 7 :9–20, 1991.

Navigation à partir d'une mémoire d'images

Ce manuscrit traite de la navigation autonome de systèmes robotiques dotés d'une caméra embarquée. Pour qu'un robot mobile puisse se déplacer automatiquement, il doit être doté de fonctionnalités lui permettant de se localiser dans son environnement, de se définir un chemin à suivre, et de contrôler ses mouvements jusqu'à sa position désirée. Dans notre approche, l'environnement de navigation est décrit par une base d'images acquise lors d'une phase d'apprentissage. L'utilisation de techniques de recherche d'images permet d'effectuer une localisation qualitative du système robotique. La définition du chemin à suivre est réalisée sans imposer une reconstruction $3D$ complète de la scène. Pour ce faire, la base d'images est organisée sous la forme d'un graphe valué dans lequel est prise en compte la facilité de mouvement entre les différentes vues de la base. Une représentation hiérarchique de ce graphe permet de simplifier la recherche de chemin. Le chemin de plus faible coût obtenu correspond à un ensemble de prises de vue permettant de relier visuellement les images acquises depuis les positions initiale et désirée du robot.

Les mouvements du robot sont calculés en ligne, pour chaque image acquise par la caméra. Le chemin d'images correspond à une délimitation de la zone de l'environnement que doit observer la caméra durant les mouvements du robot. Les différentes vues de ce chemin ne sont pas considérées comme des positions cibles intermédiaires à atteindre. Nous avons proposé une loi de commande basée sur les principes d'asservissement visuel et de fonction de potentiel, afin de contraindre le système robotique à se mouvoir dans une zone où la visibilité d'amers visuels est considérée comme satisfaisante. Les amers utilisés sont initialement détectés sur le chemin d'images, et dynamiquement mis à jour durant la navigation par une technique de transfert d'images. Les résultats expérimentaux réalisés d'une part avec un robot cartésien, et d'autre part en simulation montrent le bien-fondé de notre approche de navigation.

Mots-Clefs : Vision, Robotique, Asservissement visuel, Navigation autonome

Navigation with a visual memory

This work deals with autonomous robotic navigation. A robotic system that moves on its own must be able to localise itself in the environment, so as to define a path to follow, and also to control its motion in order to reach a desired position. A vision based-approach is chosen, by considering robotic systems with an on-board camera. The navigation space is represented in a topological way by an image database acquired during a learning step. Thus, the robot localisation is directly related to a search in the database of the views that are the closest in terms of content from the one acquired by the camera. This search is performed using image retrieval methods. A sequence of overlapping views is defined by the use of topological graphs associated with the database. These views visually describe the zone the robot must traverse to reach the desired position.

Robot motion is computed on-line for each image acquired by the camera. The proposed method is based on potential field and visual servoing principles. The control law constrains the robot to move inside an area where the observability of landmarks is considered acceptable. Features used during the visual servoing are initially detected in the path of images, and are automatically localised during the motion by using an image transfer method. Experimental results obtained with a 6 d.o.f. robotic arm, and simulations realised show the validity of the navigation method proposed.

Keywords : Vision, Robotics, Visual Servoing, Autonomous navigation