

# Table des matières

<b>Introduction</b>	<b>1</b>
<b>I Détection de fermeture de boucle</b>	<b>9</b>
<b>1 État de l’art</b>	<b>11</b>
1.1 Introduction et rappels . . . . .	12
1.1.1 Traitement de l’image et vision en robotique . . . . .	12
1.1.2 Estimation et filtrage . . . . .	13
1.2 La problématique du SLAM . . . . .	15
1.2.1 Différents types de cartes . . . . .	16
1.2.2 Différents types de capteurs . . . . .	19
1.2.3 Dérive de l’estimation . . . . .	21
1.2.4 Problématiques connexes . . . . .	22
1.3 La problématique de la détection de fermeture de boucle . . . . .	23
1.3.1 Approches métriques . . . . .	23
1.3.2 Approches de reconnaissance d’image . . . . .	29
1.3.3 Approches de classification d’image . . . . .	34
1.4 Discussion . . . . .	40
1.4.1 Information visuelle . . . . .	40
1.4.2 Les processus d’inférence . . . . .	45
1.5 Conclusion sur l’état de l’art . . . . .	47
<b>2 Cadre Bayésien pour la détection de fermeture de boucle</b>	<b>51</b>
2.1 Représentation des images et définition des lieux du modèle . . . . .	53
2.1.1 Sacs de mots visuels . . . . .	54
2.1.2 Espaces de représentation . . . . .	56
2.2 Stratégie de sélection des images . . . . .	58

2.3	Probabilité de fermeture de boucle . . . . .	60
2.3.1	Modèle d'évolution temporelle . . . . .	62
2.3.2	Système de vote pour l'estimation de la vraisemblance . . . . .	63
2.3.3	Gestion des hypothèses a posteriori . . . . .	68
2.4	Mise en cache d'hypothèses . . . . .	71
<b>3</b>	<b>Résultats expérimentaux</b>	<b>73</b>
3.1	Environnement d'intérieur . . . . .	74
3.2	Environnement d'extérieur . . . . .	80
3.3	Analyse comparative . . . . .	84
3.3.1	Influence des espaces de représentation . . . . .	84
3.3.2	Influence de la géométrie multi-vues . . . . .	86
3.3.3	Performances . . . . .	86
3.3.4	Taille des dictionnaires . . . . .	88
<b>4</b>	<b>Discussion</b>	<b>89</b>
4.1	Apprentissage du dictionnaire . . . . .	89
4.2	Gestion de l'aliasing perceptuel . . . . .	90
4.3	Caractéristiques du modèle de probabilité . . . . .	91
4.4	Conclusion . . . . .	92
<b>II</b>	<b>Application au SLAM topologique</b>	<b>93</b>
<b>5</b>	<b>État de l'art</b>	<b>95</b>
5.1	Revue générale . . . . .	96
5.2	Méthodes basées sur la vision uniquement . . . . .	97
5.3	Autres modalités perceptives . . . . .	100
5.4	Conclusion . . . . .	101
<b>6</b>	<b>SLAM topologique</b>	<b>105</b>
6.1	Structure de la carte . . . . .	106
6.1.1	Information encodée dans les noeuds . . . . .	106
6.1.2	Information encodée dans les arêtes . . . . .	108
6.2	Disposition du graphe . . . . .	110
6.3	Ajout d'une information métrique . . . . .	112
6.3.1	Relaxation pour l'estimation de la position des noeuds . . . . .	112

<b>7 Résultats expérimentaux</b>	<b>115</b>
7.1 Environnement d'intérieur . . . . .	115
7.1.1 Performances . . . . .	118
7.2 Environnement mixte . . . . .	120
7.2.1 Influence des espaces de représentation . . . . .	123
7.2.2 Performances . . . . .	125
7.3 Prise en compte de l'information d'odométrie . . . . .	127
<b>8 Discussion</b>	<b>131</b>
8.1 Perspectives . . . . .	132
8.1.1 Limites de la vision monoculaire . . . . .	132
8.1.2 Navigation . . . . .	133
8.1.3 Fusion des noeuds . . . . .	133
8.1.4 Modèle d'évolution temporelle . . . . .	133
8.2 Conclusion . . . . .	134
<b>III Application au SLAM métrique</b>	<b>135</b>
<b>9 État de l'art</b>	<b>137</b>
9.1 Revue générale . . . . .	138
9.2 Détection de fermeture de boucle dans le cadre du filtre de Kalman étendu . . . . .	140
9.2.1 Introduction d'une nouvelle estimation de pose dans le FKE . . . . .	141
9.3 Méthodes à base de filtrage particulière . . . . .	142
9.4 Méthodes indépendantes de toute solution de SLAM . . . . .	143
9.5 Conclusion . . . . .	144
<b>10 Recalage de position dans le SLAM métrique</b>	<b>145</b>
10.1 Enjeux et difficultés . . . . .	145
10.2 Aperçu de la solution mise en oeuvre . . . . .	146
10.3 Fonctionnement général de MonoSLAM . . . . .	146
10.3.1 Association de données dans MonoSLAM . . . . .	149
10.3.2 Joint Compatibility Test . . . . .	150
10.4 Détection de fermeture de boucle . . . . .	152
10.4.1 Modèle d'observation directe de la pose . . . . .	152
10.4.2 Inflation de la covariance . . . . .	154

<b>11 Résultats expérimentaux</b>	<b>157</b>
11.1 Amélioration de la procédure d'association de données . . . . .	157
11.2 Expérience de kidnapping . . . . .	158
11.3 Performances . . . . .	165
<b>12 Discussion</b>	<b>167</b>
12.1 Flexibilité et adaptabilité . . . . .	167
12.2 Correction de la carte . . . . .	168
12.3 Perspectives . . . . .	169
12.4 Conclusion . . . . .	170
<b>Conclusion</b>	<b>171</b>
<b>Bibliographie</b>	<b>179</b>

# Introduction

## Autonomie et navigation

En ce début de 21<sup>ème</sup> siècle, la place occupée par les robots dans notre société n'a jamais été aussi importante, et elle tend à s'agrandir de plus en plus rapidement dans le futur. On trouve d'ores et déjà une multitude de robots différents (cf. figure 1) dans de nombreux domaines et secteurs d'activité. Dans l'industrie par exemple, où ils sont utiles au convoi et à l'assemblage de pièces dans les chaînes de production. Dans l'exploration extra-planétaire, et notamment sur la planète Mars, où ils collectent et analysent des échantillons rocheux. Les robots font également irruption depuis peu dans la vie privée de l'Homme, chez lui, prenant des formes animales ou humanoïdes, dans le but de le distraire ou de l'assister. On observe également un intérêt notable pour la robotique dans le domaine de la Défense, où les robots deviennent de véritables suppléants, engagés par exemple dans des missions de surveillance, d'acheminement de matériel ou encore de déminage, évoluant pour cela dans différents types de milieu (terrestre, aérien, marin et sous-marin).

Dans les applications données ci-dessus à titre d'exemple, les robots peuvent être considérés comme des *agents* décidant d'actions à effectuer dans le but d'accomplir une mission. Ainsi apparaît la notion d'*autonomie* chez l'agent : pour atteindre un objectif précis, il prend des décisions et planifie la séquence d'actions correspondante. Pour être complète, cette notion d'autonomie doit également refléter la capacité de l'agent à adapter continuellement son comportement aux changements de son état et de celui de l'environnement dans lequel il évolue. Ainsi, en fonction des variations observées, de nouvelles décisions doivent être prises et de nouvelles actions planifiées afin d'atteindre le but fixé. Un agent exhibant une telle forme d'autonomie peut être considéré comme un *animat* [Meyer, 1995], [Meyer, 1996], [Meyer, 1997], et c'est de ce genre de robot dont il sera question dans les travaux rapportés ici.

La notion d'autonomie comme définie dans le contexte de l'approche animat est souvent fortement liée à la locomotion. En effet, pour atteindre son but le robot aura certainement besoin de se déplacer, afin de rejoindre un lieu en particulier par exemple. De même, lorsque le robot modifie sa trajectoire pour éviter un obstacle, il adapte son comportement pour rester en sécurité et achever sa mission. La locomotion apparaît alors comme une composante indispensable à la notion d'autonomie dans le cadre des applications

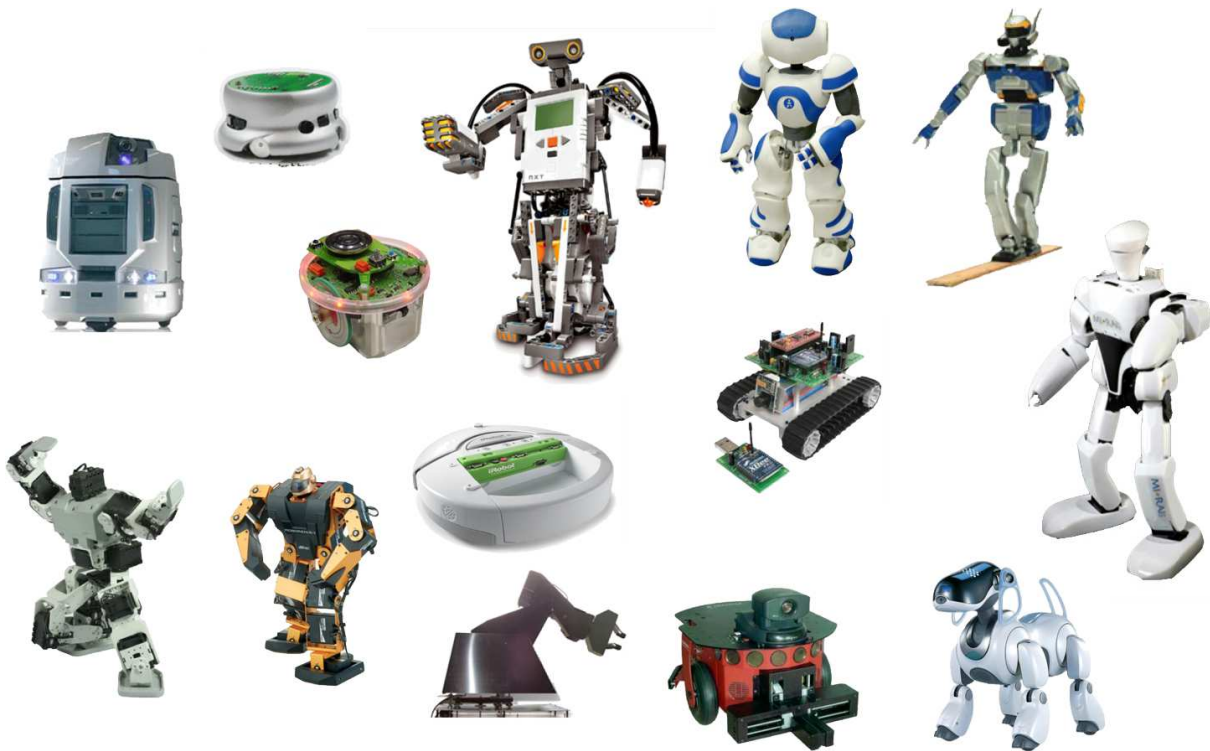


FIG. 1: Quelques exemples de robots.

de robotique. Dans ce cadre justement, lorsque la locomotion est motivée par un but quelconque, on parle de *navigation* : le robot se déplace pour accomplir une tâche spécifique. Il existe vraisemblablement chez les êtres vivants différentes stratégies de navigation, dont certaines ont même été adaptées à des applications de robotique ([Cartwright and Collet, 1987], [Gourichon et al., 2002], [Srinivasan et al., 1999]). Selon la classification proposée dans [Trullier and Meyer, 1997] et dans [Trullier et al., 1997], les stratégies de navigation employées en robotique peuvent être catégorisées en cinq grandes classes. Notamment, d'après cette classification également reprise dans [Filliat, 2001], on peut distinguer entre les stratégies avec ou sans modèle interne. Sans les présenter plus en détails, les stratégies sans modèle interne peuvent être utilisées pour réaliser des tâches telles que l'évitement d'obstacles ou l'asservissement sur une position précise. La navigation reposant sur un modèle interne permet au robot de rejoindre un but depuis des positions à partir desquelles ce but, ou les *amers*<sup>1</sup> qui caractérisent son emplacement, sont invisibles. Le modèle interne est une représentation de l'environnement qui consiste à définir des lieux comme des zones de l'espace qui sont décrites par leurs perceptions par le robot. Ce modèle peut être augmenté par l'adjonction d'informations sur les relations spatiales entre les lieux, offrant ainsi la possibilité de planifier un déplacement d'un lieu à

<sup>1</sup>Un amer est un objet, qui peut être de nature ponctuelle, dont les caractéristiques remarquables permettent de l'utiliser pour repérer la zone de l'environnement dans laquelle il se trouve.

l'autre, ou bien même par la mémorisation des positions métriques relatives des différents lieux, permettant dans ce cas au robot de planifier des chemins au sein de zones encore inexplorées. Dans ces modèles de plus haut niveau, la représentation sous-jacente consiste en une *carte* de l'environnement.

Dans le cadre des travaux rapportés ici, nous nous intéresserons exclusivement aux stratégies de navigation reposant sur une carte de l'environnement. Cela sous-entend donc qu'une telle carte soit mise à disposition, ou bien qu'elle soit construite alors que le robot découvre son environnement. Au vu de ce qui a été dit précédemment au sujet de l'autonomie et de l'adaptation au milieu opérationnel (i.e., le milieu dans lequel le robot opère), il apparaît clairement que c'est la deuxième solution qui sera préférée ici, impliquant que le robot s'acquitte de la tâche de *cartographie*. Par ailleurs, afin de pouvoir planifier un déplacement au sein cette carte, le robot doit pouvoir s'y *localiser*. En définitive, le robot doit être capable d'estimer sa position dans l'environnement tout en cartographiant celui-ci au fur et à mesure de sa progression : c'est le problème de la localisation et de la cartographie simultanées (*Simultaneous Localization And Mapping*, SLAM [Bailey and Durrant-Whyte, 2006], [Durrant-Whyte and Bailey, 2006], [Filliat and Meyer, 2003], [Meyer and Filliat, 2003], [Thrun, 2000], [Thrun, 2002]).

## Localisation et cartographie simultanées

Il existe à ce jour une multitude d'approches proposant des solutions diverses et variées au problème du SLAM. C'est l'un des premiers sujets à avoir été abordé en robotique, et ce sont les travaux de [Smith et al., 1987], ensuite mis en oeuvre plus concrètement par [Moutarlier and Chatila, 1989], qui ont permis de saisir le caractère fortement corrélé des éléments de la carte et de la position du robot : les solutions proposées jusque-là n'étaient pas satisfaisantes car elles n'avaient pas su modéliser correctement cette corrélation, réalisant alors une estimation incohérente. Depuis, plusieurs cadres théoriques ont émergé, mais tous s'attachent à rendre compte explicitement de cette propriété.

Pendant longtemps, les algorithmes de SLAM ont été associés aux "capteurs de distance" que sont les sonars, les radars et les télémètres laser. Toutefois, l'augmentation incessante de la puissance de calcul au cours de ces dernières années a permis de renverser progressivement la tendance en faveur de la vision. Ce genre de capteur était en effet jusqu'alors écarté en raison des traitements coûteux qu'il nécessite. Mais avec l'avènement de processeurs de plus en plus performants, la vision est devenue une alternative intéressante. Non seulement les caméras sont de moins en moins encombrantes et consommatrices d'énergie (facilitant ainsi leur mise en place sur des robots aux dimensions et à la charge utile réduites), mais en plus elles sont peu chères et permettent des traitements déportés grâce à la transmission des images par ondes radio. En comparaison (cf. figure 2), les capteurs de distance sont volumineux, lourds et ils nécessitent d'importantes réserves d'énergie.

De plus, l'information relative à l'apparence de l'environnement qui est encodée dans une image est beaucoup plus riche que le simple nuage de points observé grâce à un capteur de distance. Les méthodes



FIG. 2: L'utilisation de la vision permet de simplifier les dispositifs expérimentaux : une caméra monoculaire sans fil peut être facilement fixée à un ballon de baudruche gonflé à l'hélium (gauche), alors que l'utilisation d'un télémètre laser requiert une plate-forme plus imposante, dotée d'une charge utile et de réserves d'énergies conséquentes, comme en atteste le Pioneer 3DX d'ActivMedia Robotics (droite).

dérivées des domaines de la vision et du traitement de l'image permettent notamment de filtrer cette information afin d'en extraire les composantes pertinentes pour la tâche à réaliser. Ainsi, nous nous concentrerons dans le reste de ce mémoire exclusivement sur les méthodes de SLAM basées sur la vision. Les algorithmes de SLAM peuvent être classés en deux grandes catégories selon le modèle de représentation de l'environnement sous-jacent. D'un côté, les approches dites *métriques* se basent sur une représentation dense, sous la forme d'une carte dans laquelle on enregistre les positions géométriques d'amers caractéristiques de l'environnement. Dans les approches dites *topologiques*, l'environnement est segmenté sous la forme d'un graphe dont les noeuds correspondent à des lieux distincts, alors que les arêtes encodent les relations entre noeuds voisins. Cette représentation est notamment plus adaptée aux environnements de grande taille, et elle permet une navigation symbolique pour atteindre un lieu en particulier.

D'une manière générale, on observe dans les algorithmes de SLAM une dérive cumulative de l'estimation de la position (et par voie de conséquence de l'estimation de la carte) en fonction de la taille de l'environnement. Ainsi, il devient difficile dans ces conditions de pouvoir détecter les cycles dans la trajectoire du robot. Paradoxalement, lorsqu'un robot retourne sur un lieu déjà cartographié et qu'il s'en aperçoit, les estimations de sa position et de la carte pourraient être grandement améliorées, se rapprochant des quantités réelles et corrigeant en partie l'erreur accumulée. Résoudre le problème de la *détection de fermeture de boucle*, en décelant la présence de cycles dans la trajectoire du robot par la reconnaissance des lieux passés, permet donc d'améliorer la localisation et la cartographie en apportant de la robustesse aux algorithmes de SLAM.



## Détection de fermeture de boucle

La détection de fermeture de boucle peut être considérée comme une instance du problème plus général de *l'association de données* dans le cadre du SLAM, qui vise à correctement associer les mesures des capteurs obtenues à un instant donné avec les informations enregistrées dans la carte. C'est l'association de données qui permet, dans le processus de localisation du SLAM, d'inférer correctement la position du robot dans l'environnement sur la base de ses *perceptions* : l'association de données consiste donc à reconnaître dans quelle partie de la carte se trouve le robot.

Lorsque le robot ne dispose d'aucune information préalable sur sa position supposée dans la carte, trouver la bonne association de données revient à résoudre le problème de la *localisation globale*. La localisation globale peut elle-même être considérée comme un cas particulier de la détection de fermeture de boucle pour lequel on fait l'hypothèse que le robot se trouve forcément dans une partie connue de l'environnement : dans le contexte de la détection de fermeture de boucle en effet, le robot peut se trouver dans des zones encore non cartographiées. C'est par exemple ce qui est susceptible d'arriver lors d'une défaillance ou d'une obstruction des capteurs pendant un court laps de temps : une fois les perceptions rétablies, il se peut que le robot ait dérivé dans une zone qu'il ne connaît pas. Ce dernier exemple est une illustration du problème du *robot kidnappé*. Après un kidnapping, le robot peut éventuellement retourner vers une position connue, et ainsi achever un cycle.

D'après ces observations, on peut établir les contraintes basiques qui permettront de définir un cadre pratique pour résoudre le problème de la détection de fermeture de boucle. Tout d'abord, nous prétendons dans ce mémoire que la solution proposée doit être indépendante de tout algorithme de SLAM, et posséder son propre modèle de l'environnement, optimisé pour la tâche à accomplir : le but ici n'est pas d'adapter un algorithme de SLAM existant au problème de la détection de fermeture de boucle en développant une méthode avancée d'association de données. Il est pourtant indispensable que la solution proposée puisse être utilisée en conjonction avec n'importe quel algorithme de SLAM pour en améliorer les performances dans les situations énoncées plus haut. De plus, la méthode mise en oeuvre ne doit pas dépendre d'une estimation de la position fournie par un algorithme de SLAM, étant donné que, suite à un kidnapping, cette estimation sera probablement inutilisable. De même, en cas de déplacement prolongé dans une partie inconnue de l'environnement, l'incertitude sur l'estimation augmentera sans cesse, faisant de celle-ci une base peu sûre pour la détection de fermeture de boucle. Finalement, il devrait être possible d'initialiser l'algorithme produit avec une représentation de l'environnement construite par ailleurs, comme dans le cadre de la localisation globale où une carte est fournie au préalable.

## Présentation de notre contribution

Pour récapituler, la détection de fermeture de boucle est un problème d'association de données dont la résolution apporte de la robustesse aux algorithmes de SLAM en permettant de déceler la présence de cycles dans la trajectoire, de localiser le robot de manière globale (i.e., sans information a priori sur sa position), ou encore de rétablir sa position suite à la défaillance ou à l'obstruction temporaire de ses capteurs. L'intérêt de la détection de fermeture de boucle est donc majeur dans le cadre des applications de robotique, étant donné que la robustesse des algorithmes de SLAM conditionne la qualité de la navigation et, en conséquence, la capacité d'adaptation du robot. Pour résoudre ce problème, on préférera une solution indépendante, reposant sur son propre modèle de l'environnement, et pouvant être facilement combinée à tout algorithme de SLAM.

Nous proposons dans ce mémoire une méthode robuste de détection de fermeture de boucle basée sur la vision. Pour cela, nous définissons un cadre de filtrage Bayésien permettant d'estimer la probabilité que l'image perçue par le robot à un instant donné vienne d'un lieu déjà visité par le passé. L'approche mise en oeuvre a été validée à deux niveaux distincts. Tout d'abord, au niveau conceptuel, en vérifiant que les cycles apparaissant dans des séquences d'images acquises en environnement d'intérieur et d'extérieur sont correctement détectés. Ensuite, en proposant deux applications de cette méthode au problème du SLAM. La première est une implémentation directe de la méthode de détection de fermeture de boucle dans le cadre du SLAM topologique visuel, le but étant de construire une carte qui soit cohérente avec la nature cyclique de l'environnement. La deuxième application concerne la re-localisation d'une caméra dans un algorithme de SLAM métrique visuel ([Davison et al., 2004]) en cas de fermeture de boucle, mettant ainsi l'accent sur la dérive de ce genre de technique et sur l'impact positif de la correction de la position. Nous nous attachons par ailleurs également à l'amélioration de la méthode d'association de données initialement développée dans [Davison et al., 2004], afin de permettre une estimation viable même après un kidnapping de la caméra. Dans les applications évoquées ci-dessus, les expériences ont été réalisées à partir de simples caméras monoculaires, avec dans certains cas un objectif grand-angle. D'autre part, les traitements ont été effectués de façon complètement incrémentielle, au fur et à mesure que les images sont perçues lors du déplacement du robot, avec dans certains cas des performances en temps réel.

## Structure du mémoire

La première partie de ce mémoire est dédiée à la détection de fermeture de boucle. Ainsi, dans le premier chapitre de cette partie, nous dressons un état de l'art des méthodes abordant cette problématique sur la base de la vision. Cet état de l'art inclut également les méthodes de localisation globale visuelle, au vu des caractéristiques qu'elles partagent avec la problématique considérée ici. Ensuite, nous présentons en détails dans un deuxième chapitre le modèle Bayésien que nous avons développé. Le chapitre suivant est consacré à la présentation de résultats expérimentaux qualitatifs démontrant la robustesse de notre approche, sur des

séquences vidéos réalisées en intérieur et en extérieur, dans des environnements dynamiques et à l'apparence répétitive. Un dernier chapitre conclut enfin cette partie après une discussion des résultats expérimentaux obtenus.

Dans la deuxième partie, nous détaillons l'application de notre méthode au problème du SLAM topologique. Nous commençons cette partie par un état de l'art des méthodes de SLAM topologique reposant sur l'apparence. La solution que nous proposons à ce problème est ensuite abordée plus précisément dans un deuxième chapitre, détaillant notamment la structure de la carte construite et la méthode de localisation au sein de cette carte. Ensuite, une série de résultats expérimentaux est donnée dans un nouveau chapitre, afin de prouver la fiabilité de la solution développée. Encore une fois, nous terminons cette partie par un chapitre discutant les résultats obtenus avant de conclure.

La re-localisation d'une caméra dans un algorithme de SLAM visuel métrique fait également l'objet d'une partie dédiée, héritant de fait du même découpage au niveau des chapitres. Ainsi, nous brosons en premier lieu l'état de l'art des approches permettant la correction de la position de la caméra suite à une fermeture de boucle dans divers algorithmes de SLAM. Suit alors un chapitre consacré à l'exposition de la technique que nous employons pour mettre à jour la position de la caméra, ainsi que la méthode d'association de données correspondante. Le chapitre suivant donne une présentation des résultats expérimentaux obtenus dans le cadre de cette application, avant comme précédemment de discuter ces résultats et de conclure dans un dernier chapitre.

Enfin, ce mémoire est clos par une conclusion récapitulant les caractéristiques de notre modèle, en insistant sur les points forts, mais en reprenant également les limitations observées. Cela donne lieu à un ensemble de perspectives présentant les évolutions futures qu'il serait intéressant d'envisager.



## **Première partie**

# **Détection de fermeture de boucle**



# Chapitre 1

## État de l’art

L’intérêt de cet état de l’art est de présenter les méthodes basées sur la vision qui proposent une solution à la détection de fermeture de boucle. Toutefois, comme cela a été mentionné dans l’introduction générale du mémoire, la problématique de la localisation globale peut-être considérée comme un cas particulier de la détection de fermeture de boucle. Pour cette raison, les méthodes relatives à cette problématique seront évoquées ici. Par ailleurs, il ne sera pas question dans ce chapitre des techniques qui permettent de mettre à jour la position du robot et la carte associée dans les algorithmes de SLAM suite à une détection de fermeture de boucle : cela fera l’objet d’un chapitre dédié lorsque l’application correspondante sera détaillée. Ainsi, nous nous concentrerons ici exclusivement sur les aspects relatifs à la détection de fermeture de boucle et à la localisation globale, laissant pour l’instant de côté les conséquences sur les processus d’estimation des algorithmes de SLAM.

Cet état de l’art est découpé en quatre sections, plus une conclusion. Nous introduisons d’abord certaines notions indispensables à la compréhension de la suite de ce mémoire. Celles-ci concernent les domaines du traitement de l’image, de la vision en robotique, du filtrage et de l’estimation. Nous abordons ensuite de manière générale la problématique du SLAM, afin d’en définir les concepts indispensables à la compréhension de ce mémoire : cette deuxième section a pour but principal de démontrer l’intérêt de la détection de fermeture de boucle. La section suivante est dédiée à la présentation des différentes approches recensées pour la détection de fermeture de boucle et la localisation globale, selon une classification précise en fonction de leurs caractéristiques. Dans une autre section, nous discutons de ces caractéristiques en les présentant sous deux points de vue différents (i.e. l’information visuelle prise en compte et la méthode d’estimation choisie), afin d’en saisir plus évidemment les qualités et inconvénients.

## 1.1 Introduction et rappels

Cette première section a pour but d'introduire certaines notions employées dans la suite du mémoire dans les domaines du traitement de l'image et de la vision, mais également en ce qui concerne les méthodes d'estimation et de filtrage couramment utilisées dans les applications discutées par ailleurs.

### 1.1.1 Traitement de l'image et vision en robotique

Depuis l'avènement de systèmes informatiques capables de traiter l'information provenant d'une caméra vidéo, des recherches ont été menées afin d'établir des méthodes permettant d'utiliser cette information à diverses fins. En particulier, ces méthodes ont été adaptées au domaine de la robotique où les caméras tendent progressivement à devenir la modalité sensorielle de référence. Ainsi, une grande partie des concepts couramment manipulés aujourd'hui en vision de manière générale trouvent leur inspiration dans des travaux plus anciens. Par exemple, certaines méthodes de vision utiles à la navigation de robots volants [Kanade et al., 2004] sont directement dérivées de travaux [Lucas and Kanade, 1981] datant du début des années 80.

Dans la suite de ce mémoire, nous allons nous intéresser aux méthodes de vision permettant de caractériser une image. En effet, pour la tâche de la détection de fermeture de boucle, nous devons pouvoir représenter chaque image sous une forme convenable et compacte qui permette une manipulation aisée. Dans ce but, une sélection de l'information contenue dans l'image est nécessaire, la plus grande partie de celle-ci étant finalement inutile, afin de ne garder que les composantes pertinentes : ces composantes constituent les *primitives visuelles*. On peut principalement distinguer deux types de primitives visuelles : les primitives *locales* et les primitives *globales* (voir figure 1.1). Le premier type correspond à une singularité locale dans l'image, généralement un point remarquable localement par certaines de ces caractéristiques (la texture ou la couleur par exemple). Plusieurs primitives locales peuvent être extraites dans une même image (cf. notamment [Bay et al., 2006], [Harris and Stephens, 1988] et [Lowe, 2004] pour des exemples de primitives locales fréquemment employées). Dans le second cas au contraire, l'image est caractérisée par une seule et unique primitive encodant ses caractéristiques globales (cf. par exemple [Linde and Lindeberg, 2004], [Menegatti et al., 2004], ou encore [Swain and Ballard, 1991]). Dans les deux cas, on pourra employer le terme de *signature* pour désigner la représentation de l'image obtenue.

D'une manière générale, une primitive visuelle est qualifiée et identifiable par son *descripteur* : il est alors possible de comparer deux primitives en calculant une distance entre les descripteurs associés. Dans le cas d'une primitive locale, le descripteur renseignera sur le caractère remarquable de la primitive, stockant par exemple des informations de texture, de luminance ou de couleur prises dans un voisinage proche. Le descripteur identifiant une primitive globale est, quant à lui, obtenu par la collection du même genre d'information, mais sur la globalité de l'image cette fois.

Très succinctement, une représentation basée sur un ensemble de primitives locales apporte en général plus de souplesse dans la description de l'image. Cela offre une plus grande robustesse aux occultations



partielles, mais engendre par ailleurs une plus grande sensibilité au bruit local dans l'image. Il est notamment possible d'utiliser et de combiner les primitives locales de différentes manières, en fonction de la tâche visée. Par exemple, le suivi de primitives locales dans le temps (i.e., dans plusieurs images consécutives) permet d'inférer la position métrique d'amers de l'environnement pour le SLAM. Ce genre d'inférence peut également être effectué sur la base d'appariements entre les deux images d'un banc de stéréo-caméras calibré. Dans une perspective plus qualitative, les primitives locales d'une image peuvent être agglomérées dans une simple collection non-ordonnée afin de décrire la partie de l'environnement actuellement perçue. On saisit alors mieux le pouvoir à la fois quantitatif (i.e., par l'inférence de positions métriques) et qualitatif (i.e., par la collection d'information sur l'apparence de l'environnement) des primitives visuelles.

Le lecteur intéressé pourra se référer aux travaux de [Mikolajczyk and Schmid, 2003] pour une évaluation des primitives et descripteurs couramment utilisés dans les applications de vision en robotique. Par ailleurs, dans la discussion sur l'information visuelle proposée dans ce chapitre (cf. page 40), nous revenons plus en détails sur l'utilisation des différents types de primitives visuelles dans le cadre de la détection de fermeture de boucle.

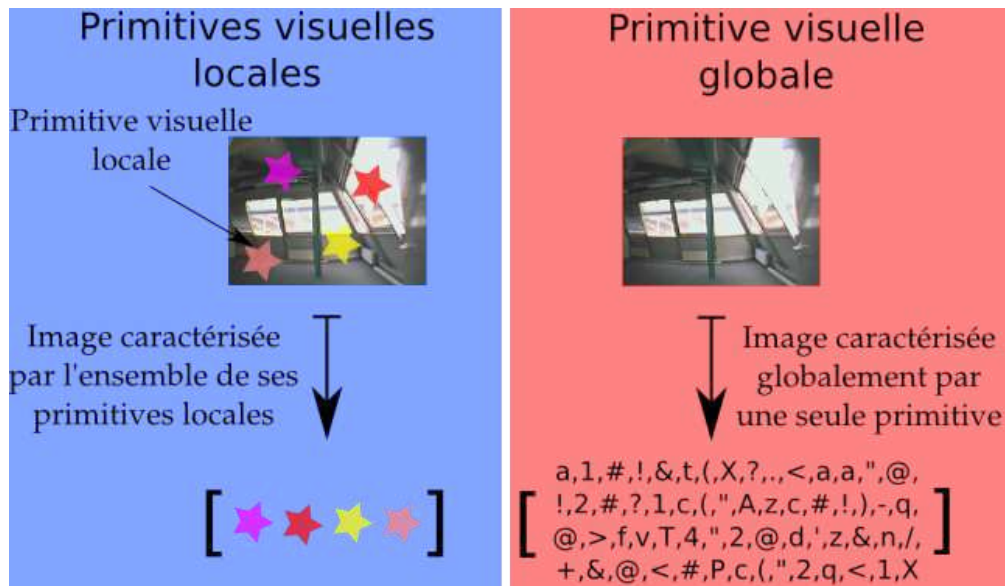


FIG. 1.1: Primitives visuelles. Une image peut être caractérisée par l'ensemble des primitives locales qu'elle contient ou bien par une seule primitive globale encodant ses caractéristiques générales.

### 1.1.2 Estimation et filtrage

Dans le cadre de la détection de fermeture de boucle et de la localisation globale, on cherche à déterminer dans quel lieu de l'environnement se trouve le robot. On peut aborder cette tâche comme un problème d'inférence dans lequel on cherche à estimer la localisation du robot. L'inférence consiste à estimer une

certaine quantité (i.e., la localisation du robot), sur la base d'observations relatives à cette quantité (i.e., les images obtenues grâce à la caméra), les différentes valeurs prises par cette quantité correspondant à autant d'hypothèses.

On peut principalement distinguer deux critères d'estimation : le *maximum de vraisemblance* (MDV) et le *maximum a posteriori* (MAP). Dans le premier cas, on cherche l'hypothèse qui semble se conformer au mieux à l'observation courante uniquement. Cela revient à mettre en correspondance les différentes hypothèses avec l'observation réalisée à un instant donné : l'hypothèse recevant le plus de crédit à cet instant sera alors considérée comme l'hypothèse la plus plausible, déterminant ainsi la valeur estimée pour la quantité inférée. Dans le cas du MAP, on cherche cette fois l'hypothèse qui semble se conformer aux mieux à toutes les observations réalisées jusque-là, et non uniquement à la dernière observation. Cela revient à intégrer l'information dans le temps, afin d'obtenir une estimation qui soit robuste aux erreurs passagères. En effet, une des principales lacunes du critère du MDV est de donner facilement du crédit à des hypothèses recevant un support trop limité dans le temps. Par exemple, lors d'une apparition soudaine et éphémère d'un objet devant la caméra, le critère du MDV favorisera certainement les hypothèses pour lesquelles cet objet était présent dans l'environnement, même si l'objet disparaît immédiatement et que la position actuelle ne correspond pas aux hypothèses sélectionnées (voir figure 1.2). Avec le critère du MAP, il faudrait que l'objet soit maintenu devant la caméra pendant un laps de temps relativement long pour qu'une localisation erronée soit décrétée.

En plus d'apporter de la robustesse en cas d'erreurs passagères, le critère du MAP permet de mieux gérer les situations d'*aliasing perceptuel* : lorsque plusieurs lieux distincts de l'environnement se ressemblent, autant d'hypothèses sont plausibles et coexistent. Dans le cas du MDV, il devient difficile dans ce genre de situation de distinguer parmi les différentes hypothèses et de sélectionner celle qui correspond à la réalité, tant elles semblent toutes également conformes à l'observation courante. Dans le cas du MAP, il est probable que l'intégration des observations au cours du temps permette de distinguer une des hypothèses, levant ainsi l'ambiguïté. Toutefois, il faut rester très vigilant dans ce genre de situation tant l'*aliasing perceptuel* peut s'avérer dangereux.

En termes d'implémentation, comme nous le mentionnons dans la discussion sur les processus d'inférence en page 45, les approches basées sur le critère du MDV sont généralement plus simples à mettre en oeuvre. Par ailleurs, en plus du modèle d'observation nécessaire à la mise en correspondance des hypothèses avec les mesures provenant des capteurs, le MAP nécessite la définition d'un modèle d'évolution temporelle pour l'intégration de l'information dans le temps. La mise en oeuvre d'un tel modèle nécessite des traitements supplémentaires, et il faut disposer d'informations sur les déplacements possibles du robot entre deux pas de temps successifs afin de définir un modèle cohérent.

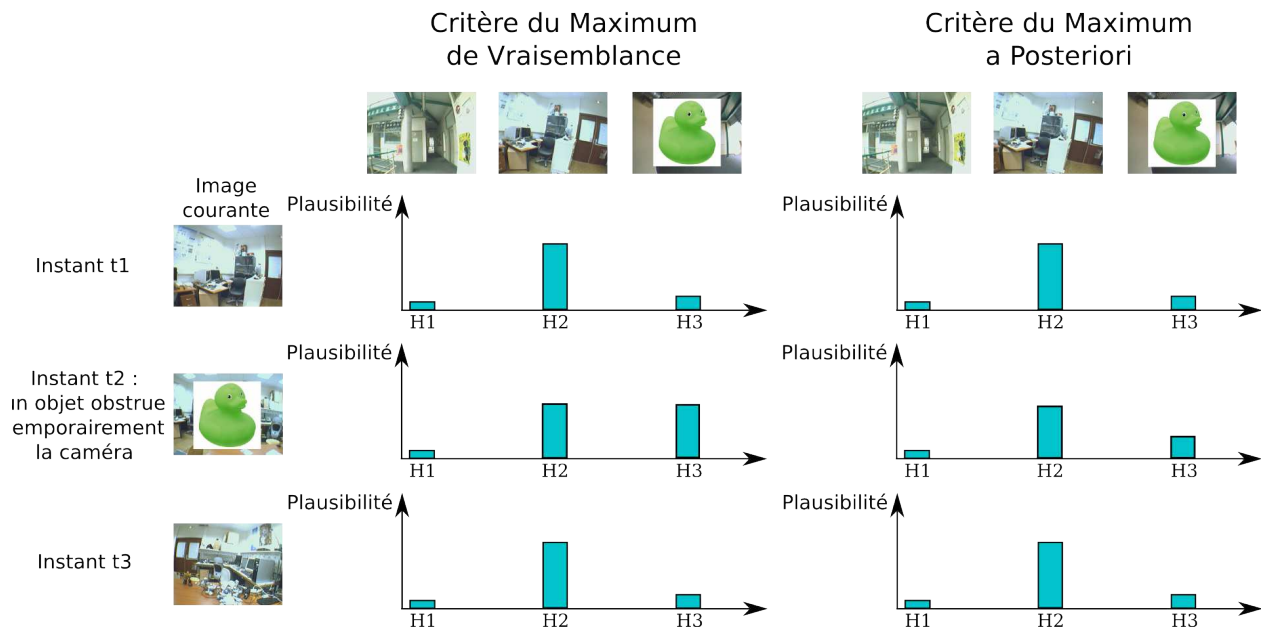


FIG. 1.2: *Maximum de Vraisemblance vs. Maximum a Posteriori. Lorsque le canard vert passe devant la caméra à l'instant t2, les hypothèses H2 et H3 du modèle de l'environnement sont tout autant plausibles l'une que l'autre au sens du Maximum de Vraisemblance : ceci est dû au fait que le canard vert était déjà passé devant la caméra dans l'image correspondant à l'hypothèse H3. Dans le cas du critère de Maximum a Posteriori au contraire, même si l'hypothèse H3 devient un peu plus plausible qu'auparavant, c'est toujours l'hypothèse H2 qui est préférée. A l'instant t3, une fois que le canard vert a disparu, tout redevient normal.*

## 1.2 La problématique du SLAM

Dans la problématique de la localisation et de la cartographie simultanées ([Bailey and Durrant-Whyte, 2006], [Durrant-Whyte and Bailey, 2006], [Thrun, 2000], [Thrun, 2002]), le robot doit inférer sa position dans une carte de l'environnement tout en construisant cette carte au fur et à mesure de sa progression. C'est notamment l'aspect concurrentiel qui rend cette tâche difficile. En effet, lorsque considérées séparément, localisation et cartographie peuvent être intuitivement facilement résolues : il semble aisé de se localiser à partir d'une carte des lieux, et la construction d'une telle carte paraît simple si on connaît sa position à chaque instant. Dans chacune de ces situations cependant, on fait l'hypothèse qu'une carte est disponible ou bien que la position est connue. Ainsi, pour résoudre le problème de la localisation, une carte est indispensable, alors que pour résoudre celui de la cartographie, il faut connaître sa position. On se rend alors compte à quel point les problèmes de localisation et de cartographie dépendent l'un de l'autre, et aborder les deux en même temps semble difficile : c'est pourquoi le problème du SLAM est souvent qualifié de problème "de l'oeuf et de la poule". Toutefois, il est possible de s'affranchir dans certaines conditions de l'étape de localisation, en employant un système de positionnement global par satellite tel que le GPS, qui permet

d'obtenir une estimation absolue de sa position à la surface de la Terre à tout instant. Ce genre de solution présente malgré tout un certain nombre de contraintes qui en limitent l'intérêt pour le problème du SLAM. D'une part, ce type de capteur ne peut pas être utilisé en intérieur, et les versions actuelles peinent à fournir une estimation précise de la position en zone urbaine. D'autre part, aucune information sur la caractérisation de l'environnement n'est disponible : cela interdit la modélisation de cet environnement sous la forme d'une carte sans l'adjonction d'un autre capteur fournissant une description des lieux visités.

C'est donc bien en considérant la localisation et la cartographie simultanément que l'on peut résoudre le problème du SLAM : l'idée générale consiste à déterminer la position courante dans la carte construite jusque-là sur la base d'une observation de l'environnement, puis de mettre à jour la carte avec les nouvelles informations obtenues à partir de cette position. Ainsi, chaque fois qu'une observation est réalisée, l'information qu'elle apporte est fusionnée avec l'estimation précédente pour en déduire une nouvelle position et une nouvelle carte.

### 1.2.1 Différents types de cartes

Nous avons déjà mentionné dans l'introduction de ce mémoire l'existence de deux types de représentation pour l'environnement : les cartes métriques et topologiques (voir notamment [Filliat and Meyer, 2003] et [Meyer and Filliat, 2003] pour une étude plus détaillée à ce sujet). Parmi les premiers formalismes proposés pour la modélisation de l'environnement, les *grilles d'occupation* ([Elfes, 1987], [Moravec, 1988], cf. figure 1.3) ont été très largement employées jusque très récemment. Dans cette approche métrique, l'environnement est découpé sous la forme d'une grille dont chaque case encode le caractère occupé ou libre d'une petite portion de l'espace grâce à une valeur binaire. D'une manière plus générale, une carte métrique fournit une représentation de l'environnement sous la forme d'amers localisés précisément par une position géométrique absolue (voir par exemple [Davison et al., 2004], [Dissanayake et al., 2001]). La position courante du robot est alors inférée à partir des positions des amers actuellement reconnus. Comme nous l'avons déjà évoqué dans l'introduction de ce mémoire, un amer est un objet dont les caractéristiques remarquables permettent de l'utiliser pour repérer la zone de l'environnement dans laquelle il se trouve. L'information élémentaire de cartographie qui constitue les amers est généralement ponctuelle (cf. figure 1.4), mais elle peut également être constituée de structures géométriques plus complexes à deux, voire trois, dimensions. Les cartes métriques sont notamment adaptées à la planification de trajectoire reposant sur un asservissement précis par rapport aux positions des amers.

La représentation proposée par les cartes topologiques est en revanche plus symbolique (voir par exemple [Booiij et al., 2007], [Cummins and Newman, 2008b], [Filliat, 2007]) : on cherche à caractériser l'apparence d'un lieu de manière générale. Pour cela, l'environnement est segmenté par les noeuds d'un graphe (cf. figure 1.5) dont les liens peuvent servir à encoder différents types d'information (adjacence temporelle, similarité, positionnement relatif etc. . .). Chaque noeud renferme alors la description qualitative d'un lieu, celle-ci pouvant prendre différentes formes en fonction principalement du type de capteur employé (voir

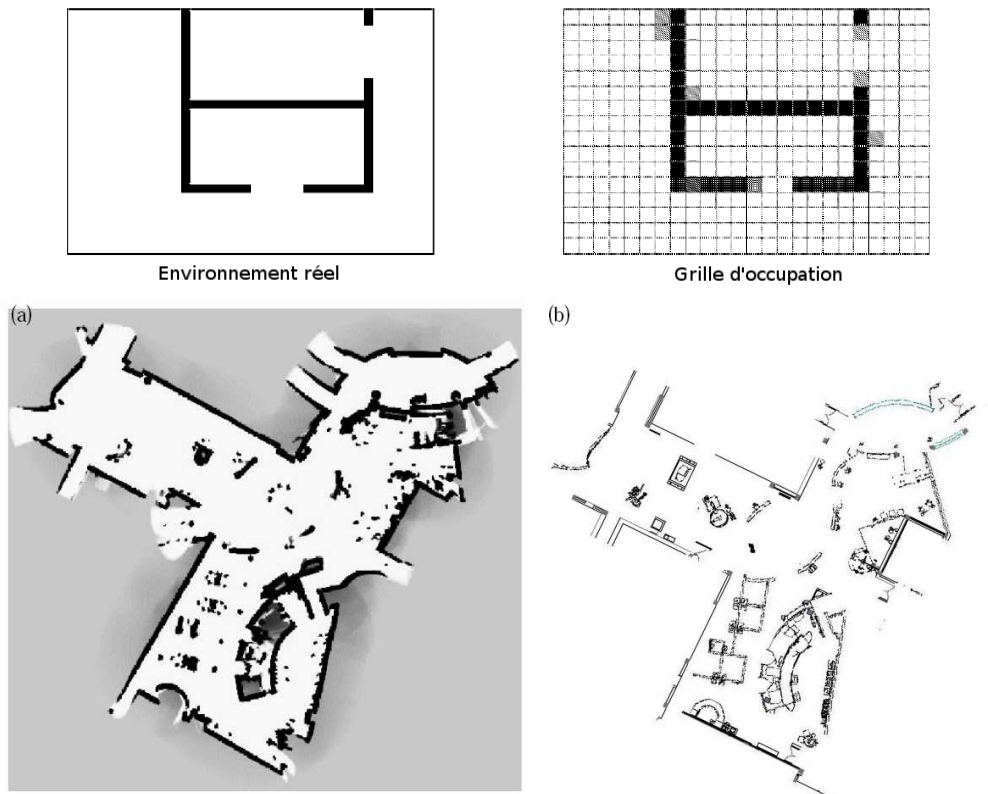


FIG. 1.3: Illustration de la représentation de l'environnement proposée par le modèle des grilles d'occupation (partie du haut, source [Filliat and Meyer, 2003]), et exemple de carte construite sur ce modèle (partie du bas, (a) la carte obtenue, (b) le plan correspondant, source : [Thrun, 2000]).

sous-section suivante et section 1.4.1 de cet état de l'art). Ainsi, déterminer la position du robot revient à trouver dans quel noeud il se trouve sur la base des mesures provenant des capteurs. La taille de la zone de l'espace encodée dans un noeud est variable, allant de la simple collection d'informations acquises à partir de positions proches, jusqu'au recensement de toutes les informations provenant d'un même endroit (une pièce dans un environnement d'intérieur par exemple). Les cartes topologiques sont généralement plus adaptées aux environnements de grande taille, et elles permettent une planification symbolique pratique pour naviguer entre les différents lieux qu'elles couvrent. Dans les approches topologiques, on peut également mentionner les solutions bio-inspirées, qui permettent de localiser le robot dans son environnement grâce aux *cellules de lieux* (voir par exemple [Filliat, 2001], [Giovannangeli et al., 2006], [Milford et al., 2004]).

Au-delà des deux formalismes généraux présentés ci-dessus pour la caractérisation de l'environnement, il existe des approches mixtes (i.e., topologique / métrique) ou encore hiérarchiques (i.e., collections de sous-cartes). Dans les approches mixtes ([Blanco et al., 2008], [Bosse et al., 2003]), on associe les deux types de représentation : l'environnement est découpé en lieux qui sont tous localement cartographiés grâce

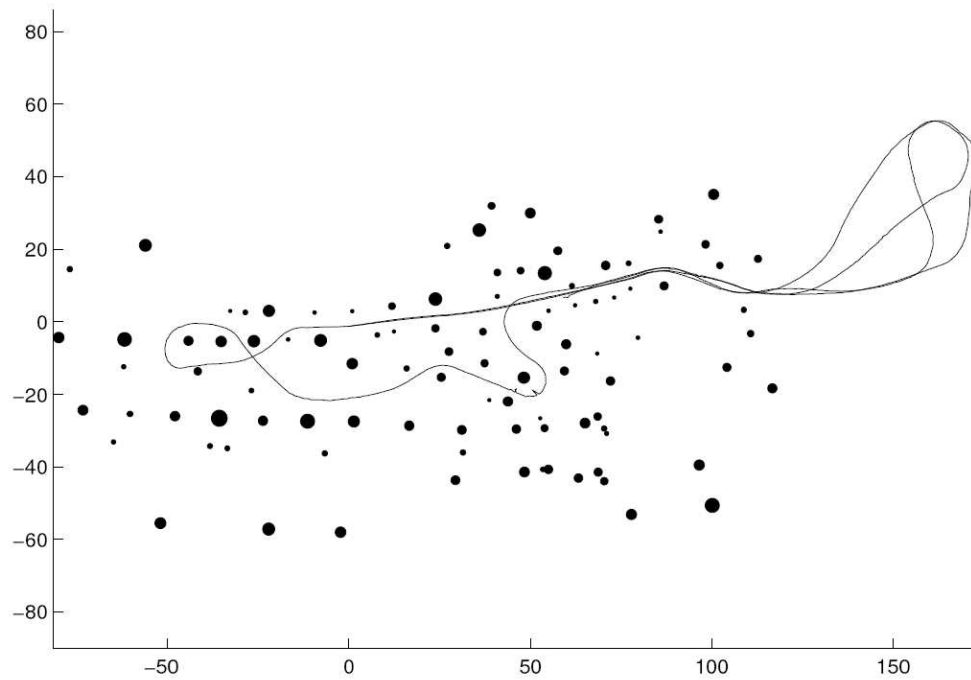


FIG. 1.4: Exemple de carte métrique 2D (vue de haut) composée d'amers ponctuels (cercles). La trajectoire d'un véhicule au sein de cette carte est également donnée. Source : [Neira et al., 2003].

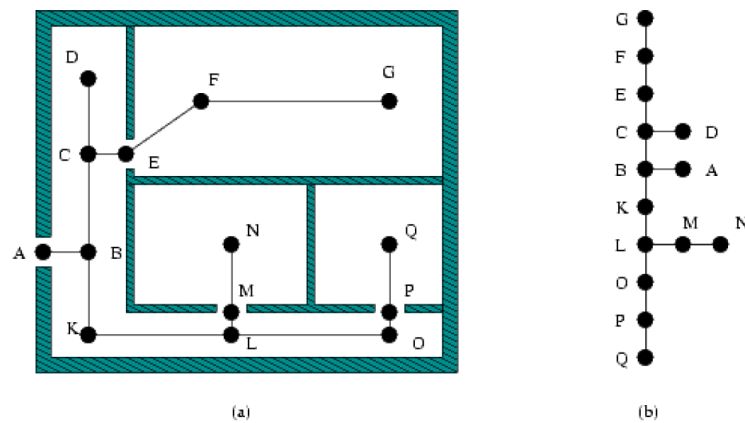


FIG. 1.5: Exemple de carte topologique 2D (vue de haut). (a) projection dans le plan. (b) Structure topologique uniquement. Source : [Dufourd, 2005].

à des approches métriques. Dans les approches hiérarchiques ([Clemente et al., 2007], [Estrada et al., 2005], [Se et al., 2005]), l'environnement est découpé en un ensemble de sous-cartes de taille limitée, bornant de cette manière la complexité du processus d'estimation.

### 1.2.2 Différents types de capteurs

On peut distinguer deux classes de capteurs pour aborder la problématique du SLAM. D'une part, les capteurs de distance tels que les télémètres laser, les radars ou les sonars, qui ont été historiquement très largement utilisés, notamment dans le cadre des grilles d'occupation. D'autre part, la vision, qui semble être devenue la modalité sensorielle de préférence depuis peu pour le SLAM. Nous avons déjà évoqué dans l'introduction de ce mémoire les caractéristiques opposant ces deux types de capteurs sur le plan pratique : une simple caméra représente un dispositif expérimental facile à mettre en oeuvre sur différents types de prototypes. Par ailleurs, l'information visuelle riche permet de caractériser l'environnement de façon qualitative (voir section 1.1.1 de cet état de l'art), alors que les capteurs de distance ne renvoient qu'une représentation de la vue courante sous la forme d'un nuage de points (voir figure 1.6). Cependant, les capteurs de distance offrent une estimation de la direction et de la distance des objets perçus, alors que la projection de la scène courante dans le plan image d'une caméra annihile toute information métrique directement utilisable. Ainsi, pour estimer la distance aux objets, il faut disposer d'un système de stéréo-caméras calibré, ou alors intégrer l'information dans le temps pour essayer de déterminer la profondeur des scènes. Dans ce second cas, il faudra certainement initialiser le système par l'observation d'une mire aux dimensions connues pour extraire le facteur d'échelle réel. Sans cela, la géométrie sera reconstruite au facteur d'échelle près.

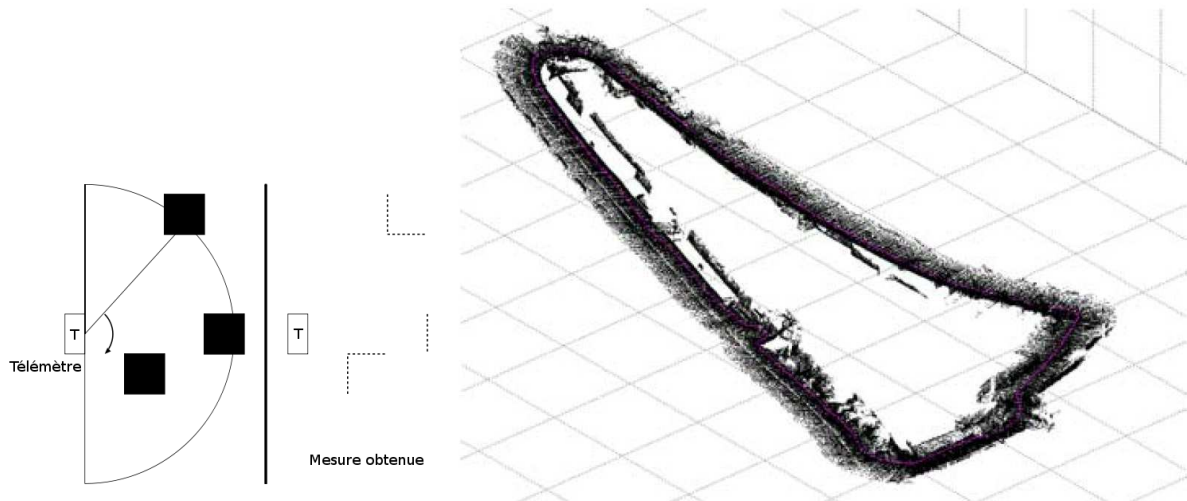


FIG. 1.6: Illustration du type d'information renvoyée par un télémètre laser (gauche, source [Filliat, 2005]) et exemple de carte 3D obtenue grâce à un télémètre laser (droite, source [Ho and Newman, 2007]). Les télémètres laser utilisent un faisceau laser mis en rotation afin de balayer un plan, en général horizontal, et qui permet de mesurer la distance des objets qui intersectent ce plan. Cette mesure peut-être réalisée selon différentes techniques (mesure du temps de retour, interférométrie...). En balançant le télémètre de manière régulière sur l'axe perpendiculaire à son plan de mesure, on peut obtenir une mesure en 3D.



Dans le domaine de la vision, il existe différents types de caméras qui peuvent être employées pour résoudre le problème du SLAM. Par exemple, l'association d'au moins deux caméras monoculaires peut servir à inférer la profondeur des scènes perçues, comme mentionné ci-dessus. Par ailleurs, les caméras omnidirectionnelles et les systèmes de caméra panoramiques permettent d'obtenir une vue à 360° de l'environnement (voir figure 1.7). Ce genre de capteur est notamment très utilisé dans le cadre du SLAM topologique, car il offre la possibilité de facilement couvrir un lieu par une seule vue. Ainsi, on peut aisément reconnaître un même lieu à partir de points de vue distants.



FIG. 1.7: Exemples d'image panoramique (haut, source [Ranganathan et al., 2006]) et d'image omni-directionnelle (bas, source [Ulrich and Nourbakhsh, 2000]).

Afin d'optimiser les traitements sur les images, les méthodes de SLAM basées sur la vision reposent généralement sur une information minimale pour caractériser l'environnement. Cela est particulièrement vérifiable dans les approches construisant une carte métrique de l'environnement à partir d'amers ponctuels : la complexité du processus d'estimation étant généralement dépendante du nombre d'éléments enregistrés dans la carte, il est primordial de limiter la quantité d'information nécessaire à l'identification des amers. Par exemple, dans les travaux de [Davison et al., 2004], chaque amer est simplement représenté par une imagerie de dimension 11x11 pixels. À partir de cette information uniquement, la détection de fermeture de boucle semble difficile. En reposant sur un modèle de l'environnement optimisé pour résoudre ce problème, il est possible de prendre en compte une information de plus haut niveau et en plus grande quantité pour assurer un taux de réussite plus élevé (voir figure 1.8).



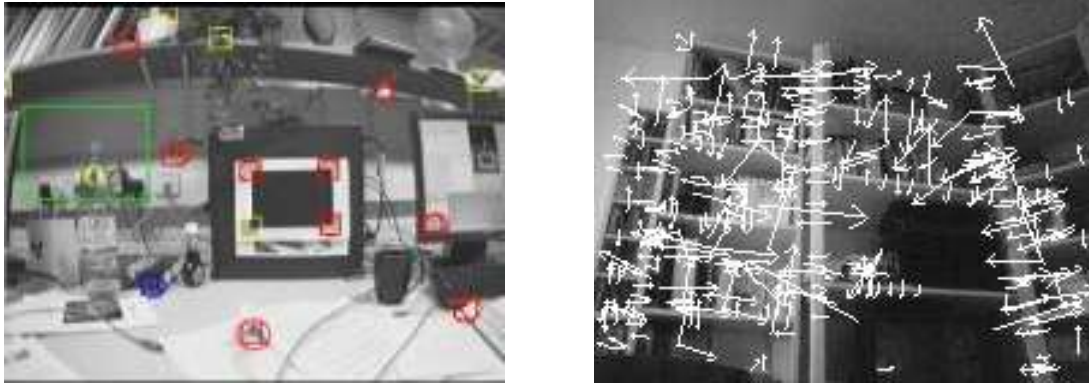


FIG. 1.8: Dans l'image de gauche sont extraites de simples imagerie de taille 11x11 pixels (carrés jaunes et rouges), alors que dans l'image de droite, des primitives de plus haut niveau (i.e., SIFT, voir section 1.4.1) sont détectées : grâce à leur descripteur plus détaillé et leur nombre plus élevé, ces primitives représentent une source d'information plus fiable pour la détection de fermeture de boucle.

### 1.2.3 Dérive de l'estimation

Plusieurs formalismes probabilistes ont été développés à ce jour pour permettre une estimation cohérente de la position du robot et de la carte de l'environnement. Cependant, on observe dans tous les cas une dérive de cette estimation lorsque le robot découvre continuellement de nouvelles zones de l'environnement, sans revenir sur ses positions antérieures. Même si la précision des solutions mises en oeuvre est constamment améliorée, rendant par exemple possible la cartographie métrique de parcours réalisés sur plusieurs centaines de mètres ([Konolige, 2004], [Thrun and Montemerlo, 2006]), il semble que l'accumulation de l'incertitude liée à l'estimation aille de pair avec la découverte de l'environnement : sans visiter à nouveau des lieux connus, il est impossible d'empêcher cette accumulation. Ainsi, dans ce genre de situation, les valeurs des quantités estimées (i.e., la position et la carte) s'écartent de plus en plus des valeurs réelles. Des exemples de cette divergence sont donnés dans les figures 1.9 et 1.10 : l'erreur accumulée au cours du déplacement du robot rend la carte incohérente, entraînant en général la duplication d'une partie de l'information qu'elle contient. Grâce à la détection de fermeture de boucle, une partie de cette erreur peut être corrigée, améliorant au final la carte construite.

Les figures 1.9 et 1.10 présentées ici illustrent bien l'importance de la détection de fermeture de boucle : on se rend compte à quel point la correction apportée permet de rétablir la cohérence des quantités estimées. Ceci est d'une importance capitale pour la suite des décisions et des traitements qui seront basés sur la carte : pour la planification et la navigation, il faut que cette carte corresponde au mieux à la structure de l'environnement. Sans cela, l'information qu'elle contient ne pourra être utilisée, rendant au final tout le processus de SLAM inutile.

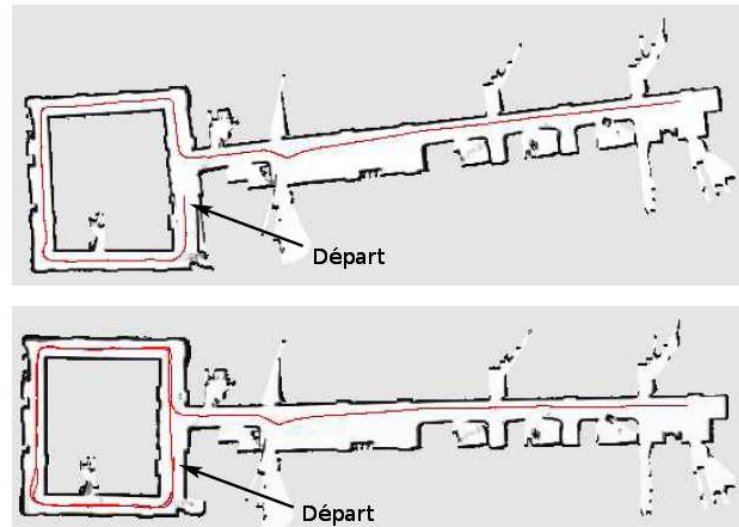


FIG. 1.9: Exemple de correction de la carte et de la trajectoire suite à une détection de fermeture de boucle : la simple reconnaissance d'une position passée permet de rétablir l'orientation de toute la partie droite de la carte. Source : [Stachniss et al., 2004]).

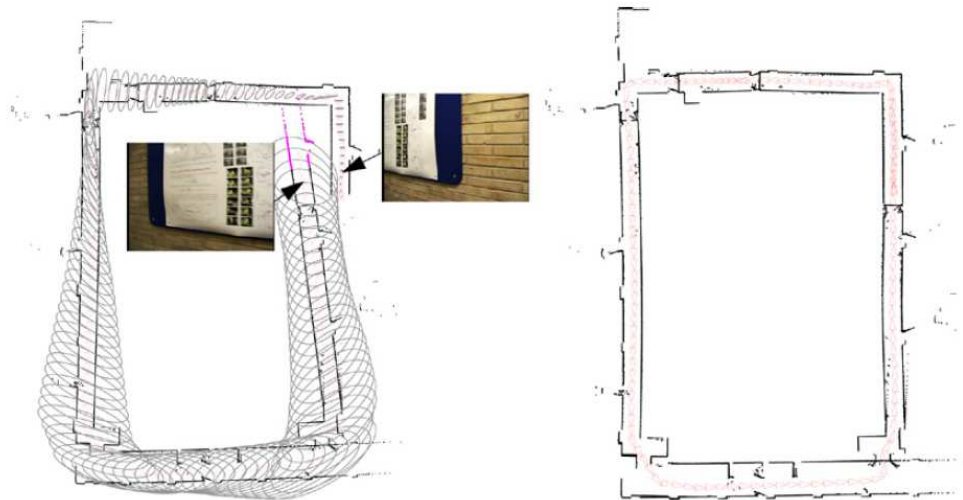


FIG. 1.10: Autre exemple de correction de la carte et de la trajectoire suite à une détection de fermeture de boucle : ici la détection de fermeture de boucle est réalisée sur la base de la vision, alors que le processus de SLAM repose sur un télémètre laser. Source : [Ho and Newman, 2006].

## 1.2.4 Problématiques connexes

Dans l'introduction de ce mémoire, nous avons mentionné des situations liées au contexte du SLAM qui étaient susceptibles d'être abordées en partie à la manière d'une tâche de détection de fermeture de boucle.

En effet, la localisation globale, qui consiste à retrouver la position du robot dans une carte construite au préalable, est un problème qui requiert la reconnaissance de lieux passés sur la base des mesures actuelles du robot (voir figure 1.11). D'autre part, lorsque le robot est kidnappé, il est déplacé dans l'environnement sans avoir connaissance des mouvements effectués, puis il se retrouve dans une zone qu'il doit tenter de reconnaître s'il l'a déjà traversée, ou bien de cartographier dans le cas contraire (voir figure 1.12). Ainsi, tant pour s'affranchir du problème de la localisation globale que pour récupérer d'un kidnapping, il faut pouvoir reconnaître des lieux passés et tenter de s'y localiser sans disposer d'aucune information sur la position actuelle. C'est précisément ce manque d'information qui rend cette tâche de localisation plus difficile que celle réalisée dans le cadre du SLAM : dans ce contexte, la dernière estimation de position sert à réduire le nombre d'hypothèses plausibles pour l'association de données (i.e., on dispose d'une prédiction de position que l'on cherche à affiner sur la base des observations courantes). Lorsqu'il faut localiser le robot sans informations a priori, toutes les hypothèses de position (i.e., sur la totalité de la carte) doivent être prises en compte, ce qui peut conduire à des situations ambiguës lorsque plusieurs lieux distincts se ressemblent. Cette phase de localisation nécessite donc la reconnaissance robuste de lieux passés, comme c'est le cas dans le cadre de la détection de fermeture de boucle. Par ailleurs, alors que pour la localisation globale on sait que le robot se trouve forcément en un lieu connu, ce n'est pas nécessairement le cas lorsque le robot a été kidnappé, ce qui ajoute une difficulté supplémentaire. En effet, comme pour la détection de fermeture de boucle, le robot peut être dans une zone de l'environnement qu'il ne connaît pas encore. Dans cette situation, il faut être capable de distinguer cette zone des autres, et ce même en présence d'une forte ressemblance avec un lieu déjà visité.

Les problèmes de la localisation globale et du robot kidnappé sont par ailleurs fortement liés au cadre du SLAM. La carte employée pour la localisation globale est généralement obtenue grâce à un algorithme de SLAM, et la méthode de filtrage mise en oeuvre pour retrouver la position du robot est souvent dérivée d'une méthode de SLAM. La phase de caractérisation de la zone de l'environnement dans laquelle se trouve le robot après un kidnapping concerne elle aussi la localisation, et les expériences de kidnapping peuvent être réalisées alors que le robot établit une carte des lieux.

### 1.3 La problématique de la détection de fermeture de boucle

Dans cette section, nous proposons une classification des différentes approches recensées pour la détection de fermeture de boucle et la localisation globale en fonction de leurs caractéristiques. Nous distinguons notamment ces approches sur la base du modèle de l'environnement sur lequel elles reposent.

#### 1.3.1 Approches métriques

Dans cette section, nous considérons en détails les approches qui reposent sur une carte métrique d'amers de l'environnement, et qui sont par conséquent généralement fortement couplées à une méthode de SLAM.

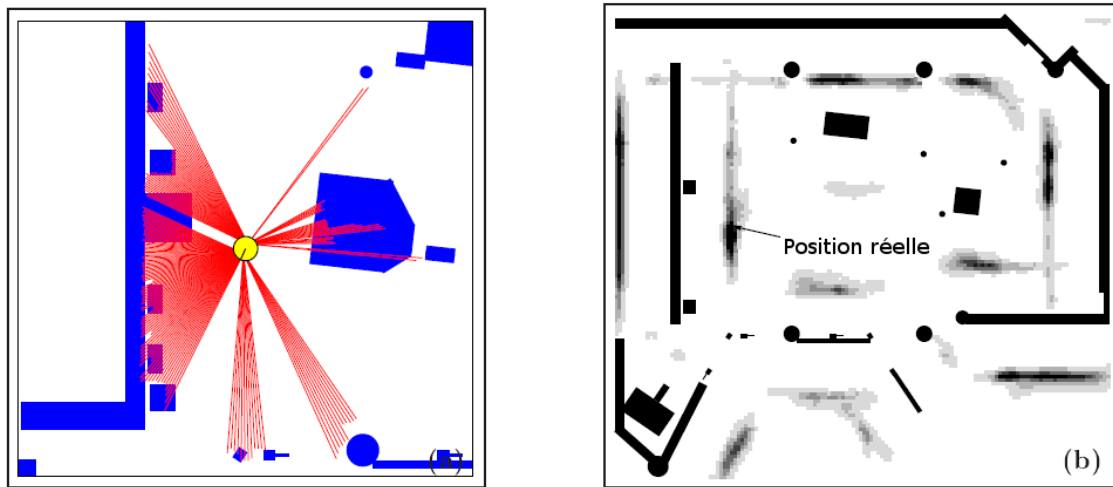


FIG. 1.11: Illustration du problème de la localisation globale : à partir des mesures provenant des capteurs (a), le robot doit inférer sa position dans la carte (b). Plus les positions sont plausibles, plus le niveau de gris est foncé. A partir d'une seule mesure il est difficile de reconnaître précisément le lieu où se trouve le robot, plusieurs positions distantes apparaissent donc comme plausibles. Source : [Fox, 1998].

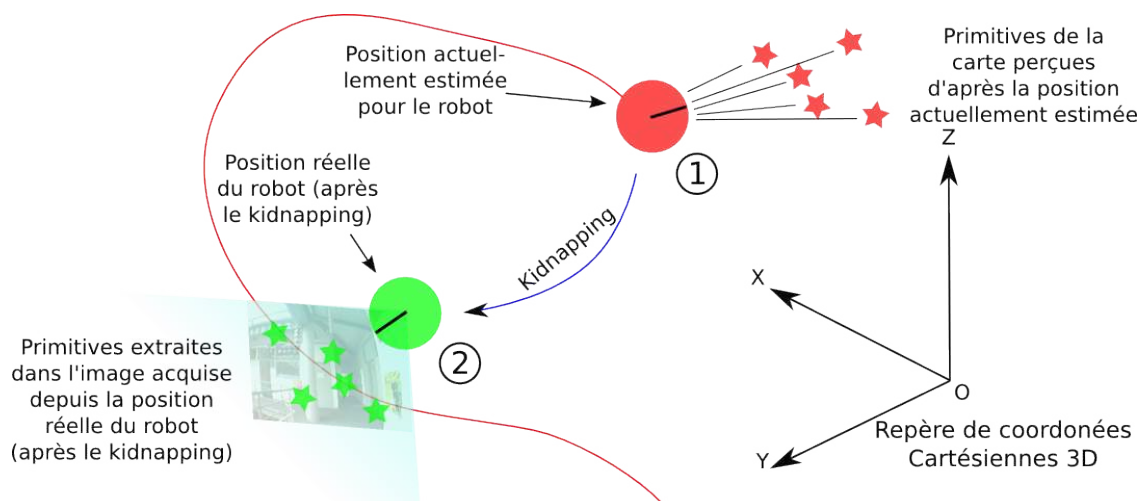


FIG. 1.12: Illustration du problème du robot kidnappé. Alors que le robot pense être à la position (1), il est kidnappé pour être déplacé à la position (2). Il faut alors mettre en correspondance les primitives extraites dans l'image acquise depuis la position (2) avec celles de la carte pour retrouver la bonne position. On peut notamment remarquer que les primitives localisées aux alentours de la position avant kidnapping ne sont plus cohérentes avec la nouvelle position.

Un amer est en fait une primitive visuelle localisée précisément par une position absolue dans la carte. Dans la plupart des approches abordées ici, reconnaître les lieux déjà cartographiés consiste en une comparaison

des primitives locales de l'image courante avec les amers de la carte (voir figure 1.13). Cette mise en correspondance pourra faire intervenir la description qualitative des amers et des primitives aussi bien que leurs positions géométriques (i.e., par projection de l'amer dans l'image ou bien par inférence de la position de la primitive dans la carte).

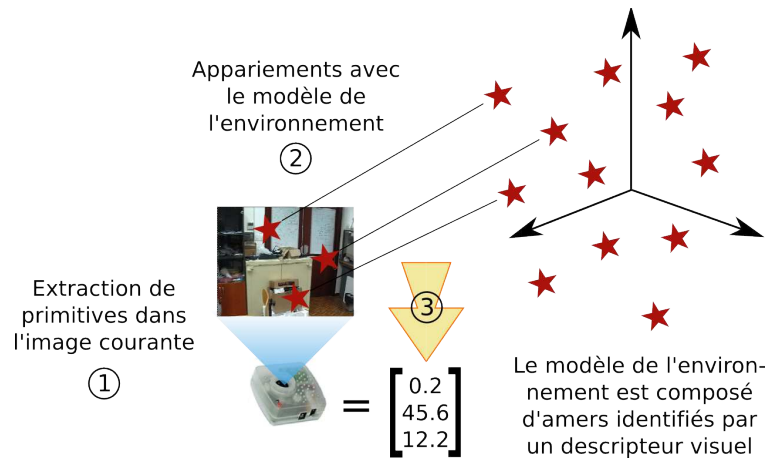


FIG. 1.13: Modèle de l'environnement pour les approches métriques : les primitives visuelles extraites dans l'image courante (1) sont comparées aux amers de la carte de l'environnement (2) pour en déduire la position métrique de la caméra (3).

Une implémentation directe de ce concept d'appariement entre les primitives locales et la carte est proposée par les auteurs de [Se et al., 2002] dans le cadre du problème de la localisation globale : la position du robot est obtenue par reconstruction 3D à partir des positions des amers qui ressemblent le plus aux primitives de l'image courante. Pour une plus grande efficacité, une procédure RANSAC [Fischler and Bolles, 1981] est employée afin de trouver rapidement un sous-ensemble d'amers de la carte qui permette une reconstruction cohérente. Cette approche, basée sur le critère du maximum de vraisemblance (MDV), repose donc essentiellement sur la robustesse de l'association de données entre les primitives courantes et les amers de la carte.

Une approche similaire [Williams et al., 2007b] est mise en oeuvre dans un algorithme de SLAM afin de retrouver la position de la caméra lorsque son suivi est interrompu (suite à une occultation du capteur ou à un mouvement brutal par exemple). Pour cela, on recherche des triplets d'amers de la carte qui ressemblent aux primitives extraites dans l'image courante. A partir d'un triplet d'amers aux coordonnées 3D, il est possible d'inférer une position métrique précise pour la caméra. La méthode d'inférence retenue ici (i.e., l'algorithme des "trois points" [Fischler and Bolles, 1981]) est implémentée dans le cadre d'une procédure RANSAC visant à générer plusieurs hypothèses de localisation : l'hypothèse qui permet d'assurer un maximum de projections cohérentes d'amers dans l'image est finalement retenue. Afin d'assurer un fonctionnement en temps réel, l'approche décrite ci-dessus a été optimisée [Williams et al., 2007a] grâce à

l'utilisation des *Randomized Tree* (RT, [Lepetit and Fua, 2006]) pour la reconnaissance des triplets d'amers présents dans l'image courante. Les RT reposent sur une forêt d'arbres de décisions binaires pour la reconnaissance de primitives visuelles. Il s'agit d'une méthode d'apprentissage nécessitant d'entraîner des classeurs (i.e., les arbres de décision) sur la base d'exemples obtenus au préalable. Dans les travaux de [Williams et al., 2007a], l'apprentissage est réalisé en ligne : lors de l'ajout d'une nouvelle primitive, on génère des exemples d'entraînement en la déformant artificiellement de 400 manières différentes. Récemment, cette approche de localisation suite à un kidnapping a été adaptée au cadre plus général de la détection de fermeture de boucle [Williams et al., 2008] : plutôt que de chercher à localiser la caméra dans une partie connue de l'environnement uniquement lorsque le suivi de position est interrompu, une telle procédure est mise en oeuvre périodiquement. Ainsi, à chaque nouvelle acquisition, on cherche à apparier les primitives de l'image courante avec des zones éloignées dans la carte, reposant pour cela sur le "graphe de co-visibilité" des amers. Ce graphe, construit de manière incrémentielle, lie entre eux les amers qui ont été observés simultanément dans la même image. Pour la détection de fermeture de boucle, on cherche donc à apparier les primitives couramment perçues avec des amers distants ce ceux actuellement observés. En cas de succès, une fermeture de boucle est détectée. En dépit des nombreuses qualités de l'approche décrite dans [Williams et al., 2007a] et dans [Williams et al., 2008] (i.e., implémentation incrémentielle, traitements en temps réel à 30Hz, correction de la position de la caméra dans le cadre d'un algorithme de SLAM métrique, récupération possible suite à un kidnapping), il semble que les RT ne soient pas robustes à l'aliasing perceptuel : comme mentionné dans [Williams et al., 2008], les RT produisent un nombre élevé de faux positifs (ceux-ci étant par la suite écartés par l'algorithme des "trois points"). Toutefois, on peut se demander à quel point cette méthode peut être généralisée à des environnements présentant un fort aliasing perceptuel, comme ceux sur lesquels se basent les expériences rapportées dans le cadre de ce mémoire.

Les méthodes d'association de données proposées ci-dessus ont par ailleurs été adaptées à d'autres cadres de SLAM métrique. En effet, les travaux de [Clemente et al., 2007] et de [Se et al., 2005] abordent le problème de la détection de fermeture de boucle dans un environnement segmenté en sous-cartes locales de tailles réduites et se recouvrant faiblement. Au lieu de simplement mettre en correspondance les primitives de l'image courante avec la sous-carte locale dans laquelle la caméra est supposée être, une plus grande quantité d'information est prise en compte dans le voisinage de la position actuelle. Dans [Clemente et al., 2007], les primitives de l'image courante sont ainsi également mises en correspondance avec les amers situés dans les sous-cartes avoisinant la position estimée. Dans [Se et al., 2005], plusieurs images consécutives sont utilisées pour construire une sous-carte locale relative à la position courante et dont les amers sont comparés à l'ensemble des sous-cartes construites jusque-là. Dans les deux cas, le problème de la détection de fermeture de boucle devient un problème de détection de recouvrement entre sous-cartes (voir figure 1.14). En particulier, l'algorithme GCB (Geometric Constraints Branch and Bound, [Neira et al., 2003]) employé dans [Clemente et al., 2007] permet d'assurer la validité de l'association de données en cas de fermeture de boucle : les données appariées doivent vérifier des contraintes unaires (i.e., critère de ressemblance) et

binaires (i.e., critère de positionnement relatif) avant d'être soumises à un test (*Joint Compatibility Test*, [Neira and Tardós, 2001]) validant la cohérence simultanée des appariements.

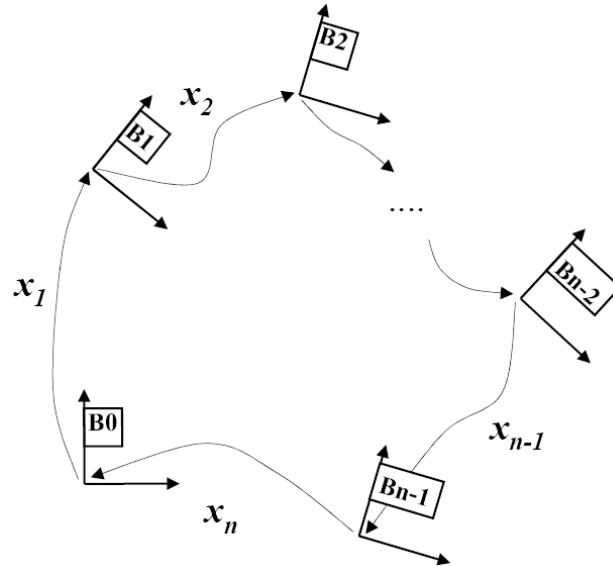


FIG. 1.14: Détection de fermeture de boucles dans le cadre d'une approche hiérarchique : par l'appariement de primitives entre les cartes  $B_{n-1}$  et  $B_0$ , on peut exprimer la position relative  $x_n$  du référentiel de  $B_0$  dans le référentiel de  $B_{n-1}$ , et ainsi détecter une fermeture de boucle entre ces cartes. Source : [Estrada et al., 2005].

Afin de garantir une plus grande robustesse face à l'aliasing perceptuel (i.e., lorsque plusieurs lieux distincts se ressemblent), l'association de données peut être réalisée dans un cadre probabiliste de filtrage particulier, afin d'intégrer l'information au cours du temps. Le modèle d'estimation sous-jacent repose alors sur le critère du maximum a posteriori (MAP, voir section 1.1.2). Ce genre de modèle a notamment été appliqué avec succès au problème de la localisation globale ("Monte-Carlo Localization", MCL, [Dellaert et al., 1999b]) et du SLAM ("Rao-Blackellised particle filter", RBpf, [Montemerlo et al., 2003]). On trouve également depuis peu des versions basées sur la vision de MCL ([Andreasson et al., 2005], [Dellaert et al., 1999a], [Wolf et al., 2005]) et de RBpf ([Barfoot, 2005], [Eade and Drummond, 2006], [Elinas et al., 2006], [Karlsson et al., 2005], [Pupilli and Calway, 2006], [Sim et al., 2005]). Le principe de base du cadre probabiliste commun aux applications listées ci-dessus consiste à approximer la distribution de probabilité de la position du robot dans son environnement grâce à un ensemble fini d'échantillons appelés *particules* (une particule correspond à une hypothèse de position), selon un processus récursif de tirages aléatoires avec remise (i.e., *sampling-importance-resampling*, SIR, [Doucet et al., 1998]). Lors de l'acquisition d'une nouvelle mesure en provenance des capteurs, chaque particule est pondérée en fonction la pertinence de l'hypothèse qu'elle représente face à la mesure obtenue : plus l'hypothèse est pertinente, plus son poids sera renforcé. Ce calcul de vraisemblance correspond globalement à une mesure de similarité entre les

primitives de l'image courante et les amers visibles compte tenu de la position représentée par la particule considérée. Ensuite, une étape de re-échantillonnage permet de concentrer les particules dans les lieux les plus vraisemblables (i.e., ceux pour lesquels les poids sont les plus élevés). En normalisant les poids ainsi calculés, on obtient la distribution discrète de probabilité recherchée. Avant la prochaine mesure effectuée par les capteurs, l'ensemble des particules sera déplacé relativement à un modèle d'évolution temporelle de la position du robot, sur la base de l'odométrie par exemple. Le processus de localisation globale MCL est illustré dans la figure 1.15. En reposant sur le critère du MAP, les méthodes de MCL et de RBpf présentent une certaine robustesse face au phénomène d'aliasing perceptuel. Elles sont par ailleurs très adaptées au cas "métrique" considéré dans cette section.

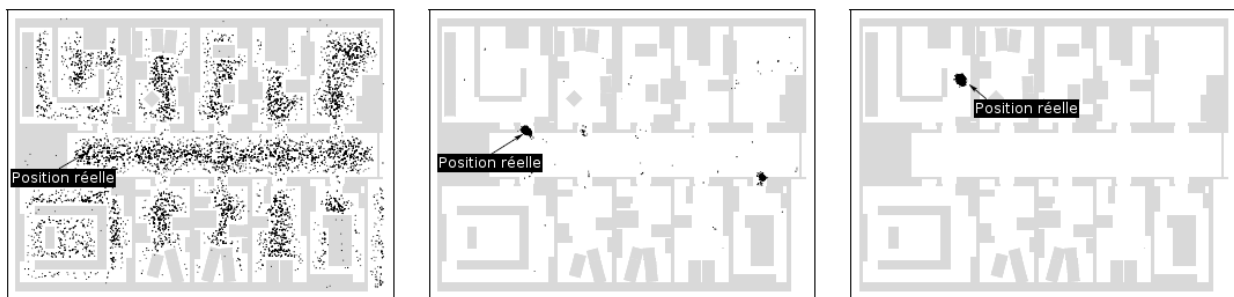


FIG. 1.15: Illustration de la convergence du processus de localisation globale MCL : au départ, les particules sont dispersées sur toute la carte, avant de se concentrer au fur et à mesure des déplacements et des observations sur la position réelle. Source : [Fox et al., 1999].

Les méthodes présentées jusque-là dans cet état de l'art pour l'association de données sont des approches dites *ascendantes* : on extrait les primitives visuelles dans l'image courante avant de les mettre en correspondance avec les amers de la carte. À l'inverse, la méthode proposée par exemple dans les travaux de [Davison et al., 2004] est dite *descendante* : les amers visibles compte tenu de l'estimation actuelle de la position sont projetés dans l'image courante avant d'être mis en correspondance avec un ensemble limité de primitives extraites aux alentours de ces projections. Selon une approche descendante également, le mécanisme d'attention visuelle mis en oeuvre par [Frintrop and Cremers, 2007] permet de prédire la présence d'un amer de la carte dans l'image courante. Pour cela, on compose successivement l'image courante avec les descripteurs d'amers proches de la position actuellement estimée. Lorsqu'un amer est effectivement présent dans l'image, la composition résulte en l'apparition d'une zone de saillance à la position de l'amer. D'après les résultats obtenus, ce mécanisme d'attention visuelle semble plus robuste aux changements de points de vue que les approches ascendantes. Il repose toutefois sur l'estimation de la position, ce qui en limite la pertinence pour les applications considérées dans ce mémoire.

Dans un cadre d'inférence probabiliste, les auteurs de [Sim and Dudek, 2004] proposent d'apprendre un modèle génératif de l'apparence de l'environnement. Grâce à ce modèle, il est alors possible d'inférer une position métrique pour le robot d'après une simple observation d'amers, sans recourir à une reconstruction



explicite à partir des positions de ces amers. Pour cela, lors d'une phase hors-ligne d'apprentissage, chaque amer de la carte est associé aux positions de la caméra à partir desquelles il a été perçu. En interpolant entre ces différentes positions, il est alors possible d'approximer la fonction liant l'observation des amers à la position de la caméra. Cette fonction, qui constitue le modèle d'observation, permet par la suite de retrouver instantanément le point de vue de la caméra à partir des amers perçus. Ce genre d'approche a pour principal défaut de reposer sur une phase d'apprentissage préalable lourde, puisqu'une importante quantité d'images prises à des points de vues proches doit être analysée pour obtenir le modèle de l'environnement.

### 1.3.2 Approches de reconnaissance d'image

Dans cette section, nous présentons les méthodes qui considèrent la détection de fermeture de boucle comme un problème de reconnaissance d'image. Cela repose sur l'hypothèse que deux images similaires proviennent probablement du même endroit. Dans les approches du type reconnaissance d'image, le lieu où se trouve le robot est déterminé en comparant l'image courante à l'ensemble des images qui constituent le modèle de l'environnement (voir figure 1.16). Pour cela, les images sont caractérisées par des primitives visuelles, locales ou globales, et comparées grâce à la distance entre descripteurs correspondante (voir section 1.1.1). La position estimée est donc "qualitative" (par opposition à la position "quantitative" obtenue avec les approches métriques de la section 1.3.1) : on cherche ici une correspondance symbolique entre images, et non une position métrique précise. Cependant, il est tout de même possible d'obtenir une information géométrique de position, en appliquant par exemple une méthode de reconstruction 3D [Hartley and Zisserman, 2004] entre l'image courante et son correspondant dans le modèle de l'environnement. Le niveau de granularité du modèle considéré ici est plus élevé que dans les approches précédentes, puisque l'entité de base n'est plus une simple primitive ponctuelle localisée précisément dans l'environnement (i.e., un amer), mais la représentation complète d'une image.

#### Comptage et vote

Une implémentation directe du type de méthode considéré ici dans le cadre de la localisation globale consiste à comparer l'image courante à toutes les images du modèle de l'environnement, afin d'en déterminer le lieu de provenance. Pour cela, il est possible de définir une mesure de similarité entre images en employant de simples méthodes de vote, comme dans les travaux de [Ulrich and Nourbakhsh, 2000] et de [Wang et al., 2006] : chaque nouvelle image acquise est mise en correspondance avec toutes les images du modèle, et son lieu de provenance est déterminé comme étant le lieu de l'image du modèle avec lequel elle partage le plus de similarités (i.e., celle qui reçoit le plus de votes). Le nombre de votes dépend ainsi de la quantité de correspondances existant entre les primitives des images comparées. Il est important de noter ici que les travaux recensés ci-dessus concernent le problème de la localisation globale : le modèle de l'environnement y est donc construit au préalable, lors d'une phase d'apprentissage supervisée et hors-ligne, par l'acquisition

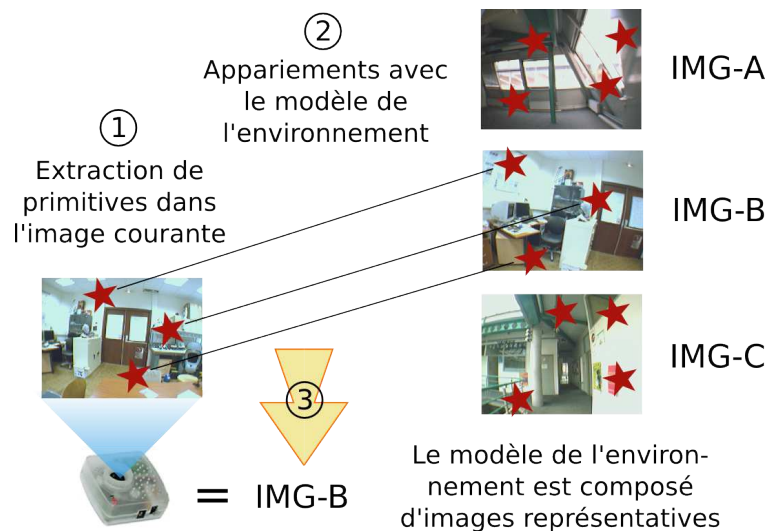


FIG. 1.16: Modèle de l'environnement pour les approches de reconnaissance d'image : les primitives visuelles extraites dans l'image courante (1) sont comparées aux primitives d'images représentatives de l'environnement (2) pour en déduire une localisation symbolique pour la caméra (3).

d'un ensemble d'images représentatives de l'environnement. Une fois ce modèle appris, il est figé et utile uniquement à des fins de localisation : on considère alors que le robot ne se déplace que dans des zones correctement représentées dans ce modèle.

Avec les méthodes de vote et de comptage recensées ci-dessus, la caractérisation des images est libre, de même que le type de caméra, à condition qu'il existe une mesure de similarité qui puissent être calculée à partir de la représentation choisie. Ainsi, les auteurs de [Ulrich and Nourbakhsh, 2000] caractérisent les images d'une caméra monoculaire avec des primitives globales, alors que dans les travaux de [Wang et al., 2006], un cadre de *sacs de mots visuels* (détaillé dans la section 1.4.1 de cet état de l'art) est employé, sur la base d'une caméra monoculaire également.

Il existe par ailleurs des approches similaires qui se placent dans le contexte de la construction de carte topologique [Booi et al., 2007] en y appliquant le même principe : un modèle de l'environnement est construit au cours d'une première phase hors-ligne, puis le robot se localise grâce au modèle obtenu. La principale différence vient du fait que dans ce cas, le modèle est appris de façon non-supervisée. Ainsi, une carte topologique est d'abord apprise à partir d'images représentatives de l'environnement, sans supervision. Pour cela, toutes les images d'entraînement sont comparées les unes aux autres de façon à les lier en fonction de leurs ressemblances : chaque arête de la carte correspond à une mesure de la similarité entre les images qu'elle relie. Cette mesure de similarité est obtenue en deux temps. D'abord, un comptage des primitives locales communes aux deux images est réalisé. Ensuite, à partir des correspondances obtenues, un algorithme de géométrie multi-vues est appliqué afin de déterminer la transformation (i.e., rotation et translation) exprimant le changement de point de vue entre ces deux images. La distance choisie ici pour

construire les arêtes de la carte topologique correspond alors à une mesure de la qualité de cette transformation (i.e., il s'agit du pourcentage de primitives locales qui peuvent être correctement projetées d'une image à l'autre compte tenu de la transformation calculée). La carte topologique ainsi construite renseigne donc sur la similarité entre noeuds : plus les images correspondantes sont semblables, plus le lien est fort. Il faut cependant remarquer que, dans les travaux de [Booi et al., 2007], une caméra omnidirectionnelle est employée et, lors de la phase d'apprentissage, la fréquence d'acquisition des images est fixée de manière à avoir un assez fort recouvrement entre images consécutives. Une fois la carte construite, le robot s'y localise grâce à la même procédure que lors de la construction de la carte, prédisant le lieu de l'image courante comme étant le noeud avec lequel elle partage la plus forte similarité (i.e., à chaque étape de localisation l'image courante est donc comparée exhaustivement à tous les noeuds de la carte).

Dans le cadre applicatif du SLAM métrique, les auteurs de [Eustice et al., 2004] et de [Lemaire et al., 2007] proposent d'implémenter de simples méthodes de comptage de similarités afin de permettre la détection de fermeture de boucle. Pour cela, l'algorithme de filtrage employé pour le SLAM dans [Eustice et al., 2004] repose explicitement sur la comparaison de l'image courante avec l'ensemble des images passées : dans le *delayed-state SLAM*, on ne construit pas de carte métrique dense d'amers, mais on estime à la place la trajectoire de la caméra à chaque acquisition d'image comme autant de points de passages, en définissant des contraintes entre ces points de passage. Ces contraintes sont obtenues par un algorithme de géométrie multi-vues et renseignent sur les transformations relatives entre images. Ainsi, la détection de fermeture de boucle revient à chercher, parmi les images passées de la trajectoire, celle qui ressemble le plus à l'image courante (sur la base d'un comptage des points communs) et avec laquelle une transformation relative peut-être calculée. Pour limiter la recherche et la rendre plus efficace, seules les images passées dont les points de vue sont proches de la position actuellement estimée sont prises en compte, rendant la détection de fermeture de boucle dépendante du processus d'estimation. Dans un cadre plus classique, les auteurs de [Lemaire et al., 2007] construisent une carte métrique d'amers de l'environnement, et maintiennent en parallèle une *mémoire visuelle* qui lie chaque amer de la carte aux images dans lesquelles il a été vu. Les images enregistrées dans cette mémoire visuelle sont par ailleurs associées à leur point de vue (tel qu'estimé par le processus de SLAM). Ainsi, l'image courante est régulièrement comparée aux images de la mémoire visuelle dont le point de vue est proche de la position actuellement estimée. En cas d'un nombre significatif d'appariements entre les primitives locales de ces images, il est possible de "forcer" l'observation des amers correspondants dans la carte pour provoquer la fermeture de boucle dans l'algorithme de SLAM. La méthode proposée est donc très simple, mais la détection de fermeture de boucle dépend de la position estimée. Par ailleurs, dans [Lemaire and Lacroix, 2007] ces travaux ont été adaptés au cadre de la vision panoramique.

Une méthode simple de SLAM topologique est mise en oeuvre dans les travaux de [Hubner and Mallot, 2007] sur la base d'images panoramiques. Chaque image est caractérisée par une primitive globale très simple, un vecteur de 72 pixels en niveaux de gris pris sur la ligne d'horizon, alors que deux images sont comparées grâce à la norme L2 séparant les vecteurs associés. Pour la construction de la carte, un noeud

décrit par l'image courante est ajouté uniquement s'il se distingue assez des autres noeuds. Pour éviter une comparaison exhaustive avec tous les noeuds de la carte, l'image courante est simplement mise en correspondance avec le dernier noeud ajouté dans la carte ou bien le dernier noeud de fermeture de boucle. D'autre part, pour détecter les fermetures de boucle, l'image courante est également comparée à tous les noeuds dont la position est proche de l'estimation courante de pose. Une telle estimation est obtenue grâce à un algorithme de relaxation contraint par les relations d'adjacence entre noeuds. Cette information de position relative des noeuds est donnée par l'odométrie du robot. Malgré les résultats positifs présentés dans [Hubner and Mallot, 2007], l'information visuelle prise en compte ici est relativement simpliste et pauvre, ce qui pourrait être problématique dans des environnements plus réalistes que l'arène de  $300 \times 300 \text{cm}^2$  sans aliasing perceptuel dans laquelle les expériences ont été réalisées.

Les auteurs de [Fraundorfer et al., 2007] se placent également dans le cadre applicatif du SLAM topologique. Dans un premier temps, une base de données d'images représentatives de l'environnement est apprise pour pouvoir ensuite caractériser les images selon le formalisme des sacs de mots visuels (voir 1.4.1 de cet état de l'art). La méthode mise en oeuvre pour cet apprentissage est similaire à celle employée dans les travaux de [Nister and Stewenius, 2006]. Une fois cette phase achevée, une carte topologique de l'environnement est inférée en-ligne au cours du déplacement du robot sur la base de la comparaison des images acquises. Pour cela, chaque nouvelle image est comparée aux images traitées jusque-là grâce à une méthode de vote pour déterminer, parmi les images passées, celle qui ressemble le plus à l'image courante. Un algorithme de géométrie multi-vues est alors employé pour infirmer ou confirmer le résultat du vote. La méthode mise en oeuvre ici est simple et efficace, mais elle repose malgré tout sur une phase préalable d'apprentissage hors-ligne.

### Cadres avancés pour l'estimation et filtrage

Dans les approches présentées jusque-là dans cette section, l'estimation du lieu de provenance de l'image courante est effectuée selon le critère du maximum de vraisemblance (MDV). Celui-ci a pour avantage principal de permettre une implémentation simple des méthodes sous-jacentes. Cependant, il présente plusieurs limitations sévères susceptibles d'être rencontrées dans des situations courantes (voir section 1.4.2). Pour palier ces limitations, divers cadres avancés pour l'estimation (i.e., des formalismes probabilistes de filtrage) ont été adaptés aux problèmes de la détection de fermeture de boucle et de la localisation globale, comme en attestent les méthodes détaillées ci-après.

Ainsi, les auteurs de [Newman et al., 2006] proposent un cadre de gestion des hypothèses au cours du temps pour assurer la cohérence temporelle de l'estimation. Pour cela, une *matrice de similarité* est construite sur la base de la comparaison de l'image courante avec l'ensemble des images traitées jusque-là. Chaque entrée  $M(i, j)$  de cette matrice stocke une valeur proportionnelle à la mesure de similarité entre les images  $i$  et  $j$ . Ainsi, en cas de fermeture de boucle, les images correspondantes formeront une suite consécutive d'entrées non-nulles sur un axe parallèle à la diagonale (voir figure 1.17). La détection de

fermeture de boucle consiste alors à extraire les éléments hors-diagonaux non-nuls de la matrice. Dans l'approche proposée ici, la détection de fermeture de boucle est réalisée en-ligne. Cependant, elle repose sur une méthode de sacs de mots visuels (détaillée dans la section 1.4.1) appliquée lors d'une phase préalable hors-ligne d'acquisition du modèle de l'environnement. Le lecteur intéressé pourra également se référer à [Ho and Newman, 2007] pour une version étendue de [Newman et al., 2006] présentant notamment des résultats complémentaires sur le problème de la mise en correspondance de cartes faites par plusieurs robots, en considérant cette tâche d'un point de vue de détection de fermeture de boucle. Par ailleurs, une matrice de similarité est également calculée dans les travaux de [Zivkovic et al., 2005] pour construire une carte topologique de l'environnement au cours d'un processus hors-ligne.

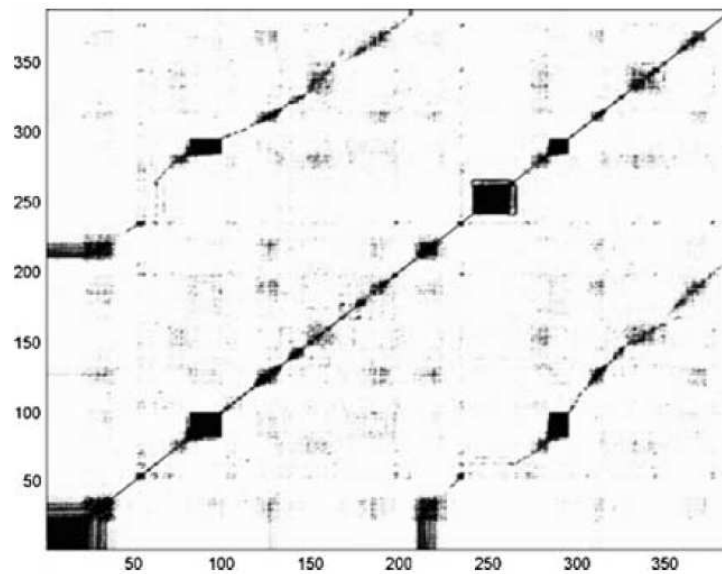


FIG. 1.17: Exemple de matrice de similarité. Chaque élément  $M(i, j)$  permet d'évaluer la similarité entre les images  $i$  et  $j$  : plus la similarité est importante, plus la cellule est foncée. Les éléments de la diagonale apparaissent logiquement avec une teinte foncée puisque toutes les images sont semblables à elles-mêmes. Les fermetures de boucles apparaissent comme les séquences connectées de cellules hors-diagonales foncées. Source : [Ho and Newman, 2007].

Afin de garantir une estimation cohérente dans un cadre probabiliste robuste, les techniques récursives de tirages aléatoires MCL et RBpf ont été adaptées au cadre de la reconnaissance d'image considéré dans cette section, comme en attestent les travaux de [Menegatti et al., 2004] (MCL) et de [Weiss et al., 2007] (RBpf). Une fois encore, la distribution de probabilité de la position du robot est discrétisée sous la forme d'un ensemble de particules pondérées. Pourtant, on ne compare plus ici les primitives de l'image courante avec les amers d'une carte métrique, mais on considère plutôt l'appariement de l'image courante avec l'ensemble des images faisant partie du modèle de l'environnement. Ce modèle est construit lors d'une phase préalable hors-ligne, au cours de laquelle chaque image considérée est associée à une position métrique

précise obtenue grâce à une vérité terrain. Plus précisément, dans [Menegatti et al., 2004] chaque image est caractérisée par une primitive globale et localisée par une position métrique absolue, alors que dans [Weiss et al., 2007], ce sont les primitives locales de chaque image qui sont localisées par leurs points de vue. A partir de ce modèle, lorsqu'une nouvelle mesure en provenance des capteurs est obtenue, on met à jour le poids de chaque hypothèse de position (i.e., chaque particule) en fonction des ressemblances trouvées entre l'image courante et les images du modèle. L'information métrique de position est alors obtenue à partir du point de vue associé à chacune des images du modèle.

Il existe également des approches qui s'attachent à définir un cadre de représentation des images optimisé en fonction des caractéristiques de l'environnement. L'idée de l'Analyse en Composantes Principales (ACP) mise en oeuvre dans les travaux de [Gaspar et al., 2000], [Kröse et al., 2002] et [Sim and Dudek, 1999] est d'apprendre, à partir d'une base de données d'images représentatives de l'environnement, une base optimale pour la projection ultérieure des images acquises par le robot. Il est important de noter ici que la base peut-être apprise à partir de toute l'information contenue dans des images omnidirectionnelles ([Gaspar et al., 2000]), à partir d'une version réduite de cette information dans le même type d'image ([Kröse et al., 2002]), ou bien même à partir de primitives extraites dans des images provenant d'une caméra monoculaire ([Sim and Dudek, 1999]). La base obtenue ne retient que les dimensions de l'espace d'entrée (i.e., l'espace de représentation des images) qui apportent une information pertinente, les autres dimensions étant négligées : le nombre final de dimensions est de ce fait limité, ce qui rend les traitements ultérieurs rapides. Plus précisément, cette base optimisée est composée des vecteurs propres les plus pertinents obtenus à partir de la matrice de covariance formée par toute l'information de l'espace d'entrée disponible dans la base d'apprentissage. Par exemple, dans [Gaspar et al., 2000], l'apprentissage étant directement réalisé sur les images omnidirectionnelles, les vecteurs propres obtenus au final sont en fait des "images propres" (voir figure 1.18). Une fois la base construite, on y projette l'image courante (ou sa représentation) pour calculer la distance qui la sépare des images (ou primitives) d'apprentissage. Pour cela, on compare simplement les coordonnées de l'image courante dans la base optimisée avec celles des images d'apprentissage. A partir de là, les auteurs de [Gaspar et al., 2000] admettent que l'image considérée vient du même lieu que l'image d'entraînement la plus proche dans l'espace de représentation optimisé. Dans le cas de [Kröse et al., 2002] et de [Sim and Dudek, 1999], un modèle génératif de l'apparence de l'environnement est également appris lors de la phase préalable d'entraînement, à partir de positions données par une vérité terrain. Cela permet par la suite d'inférer une position métrique précise pour l'image courante, sans reconstruction 3D explicite.

### 1.3.3 Approches de classification d'image

La quatrième section de cet état de l'art concerne les approches qui abordent le problème de la détection de fermeture de boucle comme une tâche de classification d'image : le but est de déterminer la classe (i.e., le lieu) de l'image courante à partir de sa description. Un lieu de l'environnement est donc décrit par un ensemble d'images, chacune correspondant à une vue de ce lieu (voir figure 1.19). Les images caractérisant

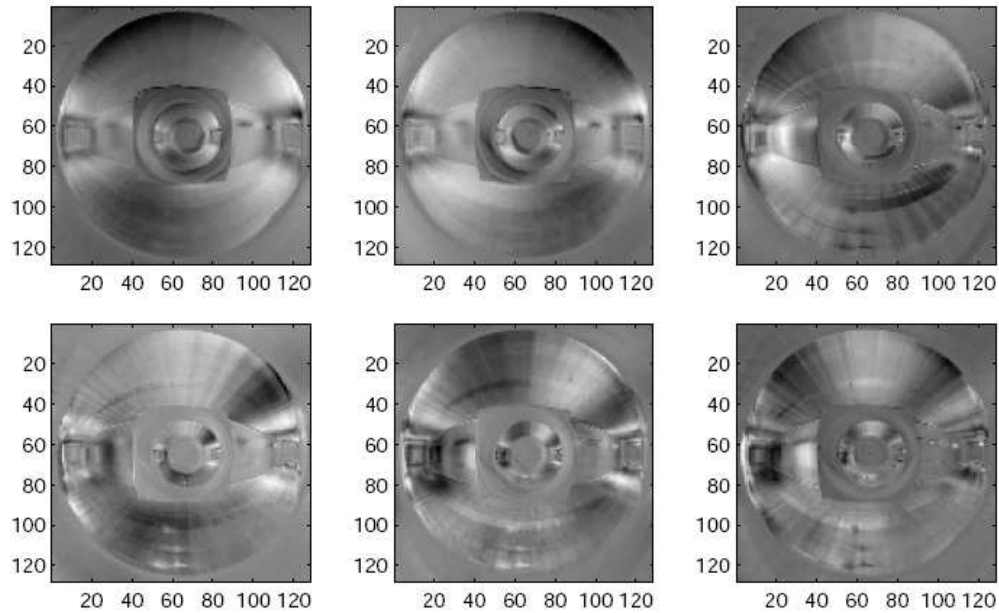


FIG. 1.18: Exemples “d’images propres” constituant la base optimisée utilisée dans les méthodes d’ACP pour la comparaison des images. Ces 6 images correspondent aux 6 vecteurs propres les plus pertinents (i.e., avec les valeurs propres les plus élevées) obtenus à partir de la matrice de covariance formée par toutes les images omnidirectionnelles de la base d’apprentissage.

un lieu sont représentées par des primitives globales ou bien par des collections de primitives locales. Les approches dont il est question ici étendent donc le cadre des approches de la section précédente dans la façon dont le modèle de l’environnement est maintenu : un lieu n’est plus seulement décrit par une seule image, mais par l’ensemble des images qui tombent dans la catégorie qu’il représente. Cette section est séparée en deux. Dans une première sous-section, nous considérons les approches pour lesquelles un lieu est défini par un ensemble d’images correspondant à des points de vue proches dans l’espace, admettant généralement un recouvrement entre ces images : il s’agit d’images identifiant visuellement la même partie de l’espace. Dans la seconde sous-section, nous détaillons des approches qui définissent les lieux de manière plus générale : chaque image caractérisant un lieu peut être acquise d’un point de vue éloigné des autres, et caractériser une zone de l’espace sans recouvrement avec les autres images. Les approches dont il est question dans cette dernière sous-section s’attachent par exemple à représenter chaque pièce d’un appartement comme un lieu distinct, alors que dans le premier cas, on identifie plutôt une portion de la trajectoire passée du robot par les images qui la composent.

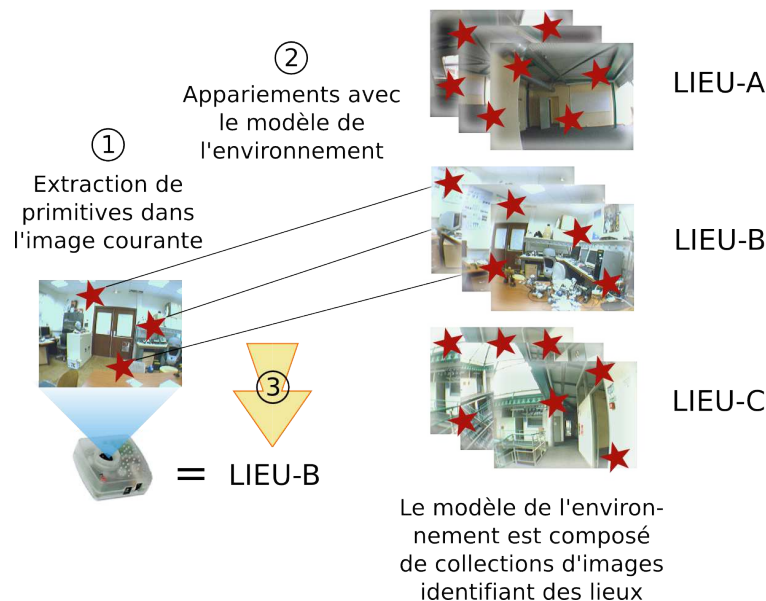


FIG. 1.19: *Modèle de l'environnement pour les approches de classification d'image : les primitives visuelles extraites dans l'image courante (1) sont comparées aux primitives d'images représentant des lieux de l'environnement (2) pour en déduire une localisation symbolique pour la caméra (3).*

### Lieux comme collection locale d'images

Les méthodes de vote et de comptage déjà rencontrées dans le cadre des approches de reconnaissance d'image ont été adaptées avec succès à la formulation du problème de la détection de fermeture de boucle considérée ici. Par exemple, dans les travaux de [Kosecká et al., 2005], chaque lieu est associé à un score renseignant sur la plausibilité de provenance de l'image courante (i.e., si la note est élevée, l'image courante a de grandes chances d'appartenir à ce lieu). Comme dans la section précédente, ce score est évalué à partir du nombre de correspondances trouvées entre les primitives locales extraites dans l'image courante et dans les images caractérisant le lieu considéré. Le critère du maximum de vraisemblance est donc encore une fois retenu pour déterminer la classe de chaque image analysée. Dans les travaux mentionnés dans ce paragraphe, le modèle de l'environnement est construit lors d'une phase préalable hors-ligne d'apprentissage supervisé. Lors de cet apprentissage, chaque lieu est défini par la collection de primitives locales extraites dans les images étiquetées comme appartenant à ce lieu. Le modèle ainsi obtenu est ensuite utilisé pour la tâche de localisation globale d'un robot.

Afin de palier les limitations du critère du maximum de vraisemblance pour la classification d'images, différents cadres d'estimation et de filtrage ont été proposés. Notamment, dans les travaux de [Goedemé et al., 2007] sur la construction de carte topologique sur la base de l'apparence uniquement, les auteurs ont choisi de modéliser les événements de fermeture de boucle grâce à un formalisme mathématique dérivé de la théorie de l'évidence. Ainsi, lors d'une première phase hors-ligne d'apprentissage non-supervisée, une



carte topologique de l'environnement est construite à partir d'une collection d'images acquises depuis une caméra omnidirectionnelle. L'acquisition est réalisée à intervalles de temps réguliers, de façon à obtenir un recouvrement partiel entre les images. Ensuite, l'algorithme de construction de carte se charge de déterminer quelles images viennent du même lieu. Pour cela, une mesure de similarité entre images, faisant intervenir à la fois des primitives globales et locales, permet de sélectionner les images qui se ressemblent (et qui viennent donc potentiellement du même lieu). Des groupes d'images sont donc ainsi définis sur la base de l'apparence uniquement. A l'intérieur de chaque groupe, il est possible de définir plusieurs sous-groupes grâce à l'information de voisinage temporel : les images d'un même groupe acquises à des instants proches dans le temps peuvent être rassemblées. Cela permet de distinguer, parmi les images qui se ressemblent, celles qui ont été acquises au même moment. Ainsi, la fusion de sous-groupes constitue une fermeture de boucle : deux sous-groupes d'images similaires prises à des instants éloignés correspondent à deux passages au même endroit à deux moments différents. Toutefois, ces sous-groupes peuvent également correspondre à deux lieux distincts qui se ressemblent. Pour lever l'ambiguïté, une méthode de collection d'évidence reposant sur la théorie de Dempster-Shafer est employée. Pour cela, on essaye de combiner les sous-groupes faisant partie d'un même groupe avant de mesurer le *support* de l'hypothèse résultante. Si celui-ci est au dessus d'un certain seuil de probabilité, la fermeture de boucle est acceptée dans la carte construite. Sinon, elle est rejetée et chacun des sous-groupes forme alors un groupe à part entière. Le processus de gestion des sous-groupes est illustré dans la figure 1.20. Une fois la carte construite, celle-ci est utilisée lors d'une seconde phase pour la localisation globale d'un robot. Cette fois, une méthode classique de filtrage Bayésien permet d'estimer la probabilité de la position du robot, sur la base de la même mesure de similarité que pour la construction de la carte.

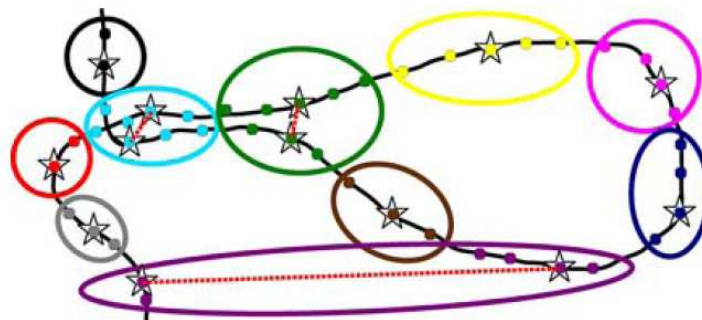


FIG. 1.20: Illustration de la méthode de regroupement d'images pour la caractérisation des lieux. Chaque ellipse regroupe les images (i.e., les points dans la figure) sur la base de la similarité uniquement. A l'intérieur de chaque ellipse, on définit des sous-groupes sur la base de la proximité temporelle d'acquisition des images. Un représentant pour chaque sous-groupe est désigné et caractérisé dans la figure par une étoile. La théorie de Dempster-Shafer permet alors de décider si un groupe doit être scindé ou fusionné. La fusion de sous-groupes correspond à une détection de fermeture de boucle. Source : [Goedemé et al., 2007].

Les auteurs de [Valgren et al., 2007] préfèrent adopter une méthode nommée *incremental spectral clustering* (ISC), afin de grouper les images provenant d'un même lieu dans le cadre du SLAM topologique. Ainsi, toute image acquise à partir d'une caméra omnidirectionnelle est comparée au représentant de chaque groupe (i.e., chaque lieu) du modèle de l'environnement obtenu jusque-là. Pour cette comparaison, une mesure de similarité basée sur des primitives locales (i.e., un simple comptage des primitives communes) est employée. Grâce à cette mesure, on peut mettre à jour une matrice d'affinité renseignant sur les similarités entre l'image courante et l'ensemble des groupes. L'algorithme ISC permet ensuite de déterminer le nombre de groupes optimal compte tenu des entrées de cette matrice, sur la base de l'apparence uniquement. L'opération résulte alors en la mise à jour d'un groupe existant avec la nouvelle image, ou bien en la création d'un nouvel ensemble de groupes. Dans tous les cas, on cherche à maximiser la proximité des images en fonction de leur apparence. Alors que l'image courante est simplement comparée aux représentants de chaque groupe (et non à chaque individu du groupe) pour optimiser les performances, tous les individus sont mémorisés, afin de réordonner les groupes si l'algorithme ISC le requiert. La méthode ainsi obtenue est incrémentielle, et elle permet de résoudre le problème du SLAM topologique sur la base de l'apparence uniquement, avec des traitements réalisés en-ligne mais avec une complexité élevée (l'algorithme ISC nécessite plusieurs décompositions en valeurs singulières de la matrice d'affinité).

Le formalisme probabiliste de filtrage Bayésien décrit dans les travaux de [Cummins and Newman, 2007] propose une solution au problème du SLAM topologique sur la base de l'apparence seulement, reposant simplement sur une caméra monoculaire. La méthode mise en oeuvre permet de déterminer la probabilité que deux images viennent du même lieu, dans le cadre d'une estimation au sens du maximum a posteriori. Les images et les lieux obtenus sont encodés par des collections de primitives locales, d'après le paradigme des sacs de mots visuels (voir section 1.4.1). Pour évaluer la probabilité de fermeture de boucle, un modèle génératif de l'apparence de l'environnement est appris lors d'une première phase hors-ligne. Au cours de cette phase, les probabilités de co-occurrence des primitives visuelles locales extraites dans des images d'entraînement sont estimées. Les images d'entraînement sont recueillies sur une vaste zone représentative du type d'environnement dans lequel le robot évoluera par la suite. A partir de ces probabilités, il est alors possible, lors d'une seconde phase d'exploitation en-ligne, de déterminer la probabilité de provenance de l'image courante. L'algorithme obtenu présente des caractéristiques remarquables, notamment une robustesse impressionnante face à l'aliasing perceptuel. On peut toutefois regretter l'aspect non-incrémentiel lié à l'apprentissage du modèle de l'environnement, ainsi que les temps de calcul importants interdisant des traitements en temps réel (une version optimisée de l'algorithme est présentée dans [Cummins and Newman, 2008b], sans toutefois atteindre des performances en temps réel).

### **Lieux comme collection globale d'images**

Dans le cadre de la localisation globale d'un robot, les auteurs de [Pronobis et al., 2006] emploient une technique de Machines à Support Vecteur (MSV) pour apprendre un classifieur qui permette de prédire le lieu

de l'image courante (voir figure 1.21). Le MSV est appris lors d'une phase hors-ligne, à partir d'une collection d'images étiquetées manuellement et décrivant un ensemble restreint de lieux (i.e., quelques pièces dans un environnement d'intérieur). L'apprentissage consiste, d'après le principe de base des MSV, à projeter les images d'entraînement dans un espace où il est possible d'opposer chacune des classes entre elles par des séparations linéaires. Afin de construire cet espace, les images sont encodées sous la forme de primitives globales. Une fois le classeur appris, on détermine la classe de l'image courante en projetant sa représentation dans l'espace de discrimination : on calcule alors la distance qui sépare cette image de l'ensemble des surfaces de séparation pour savoir dans quelle catégorie (i.e., quel lieu) elle tombe. Récemment, cette méthode a été améliorée [Pronobis and Caputo, 2007] pour pouvoir prédire la classe d'une image de façon graduelle : on traite l'image petit à petit, en essayant d'en déterminer le lieu de provenance à partir d'une information partielle uniquement. Cela permet dans certains cas d'obtenir la classe de l'image sans nécessiter l'analyse complète de celle-ci, améliorant en conséquence la rapidité des traitements. Par ailleurs, cette méthode a été également adaptée [Luo et al., 2007] pour pouvoir mettre à jour le classeur MSV avec de nouveaux exemples en-ligne, ce qui permet d'accroître la robustesse aux changements de condition dans l'environnement (illumination, objets qui ont été déplacés), sans pour autant permettre l'ajout de nouveaux lieux dans le modèle. En dépit des bonnes performances obtenues dans les résultats expérimentaux présentés dans les trois articles listés ci-dessus, l'inconvénient majeur des approches MSV est de reposer sur une phase hors-ligne pour l'apprentissage du modèle.

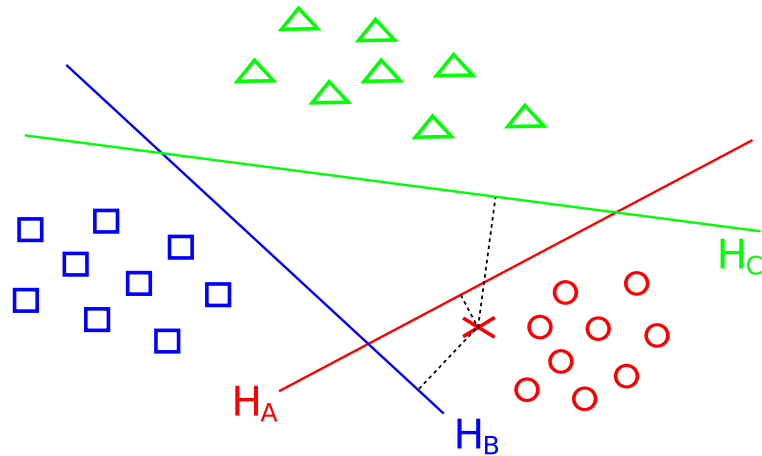


FIG. 1.21: Illustration du principe de fonctionnement des MSV. Après avoir projeté une nouvelle donnée dans l'espace de classification (i.e., la croix dans la figure), la position de cette donnée par rapport aux hyper-plans (i.e.,  $H_A$ ,  $H_B$  et  $H_C$ ) séparant les différentes classes (i.e., A, B et C) permet de déterminer à quelle catégorie elle appartient.

Dans un contexte similaire, les travaux de [Filliat, 2007] reposent sur une méthode d'apprentissage supervisée pour aborder le problème de la localisation globale qualitative. Pour cela, une méthode incrémentielle d'apprentissage basée sur l'apparence uniquement et en interaction avec un superviseur a été

développée. Cette méthode caractérise les images par des primitives locales, et les lieux par les collections de primitives locales des images correspondantes, selon une version incrémentielle du formalisme des sacs de mots visuels (c'est la caractérisation qui est implémentée dans ce mémoire également, une présentation détaillée en est donc fournie dans la section 2.1.1 du chapitre 2). Lorsque le robot acquiert une nouvelle image, il cherche à en déterminer le lieu grâce à un simple mécanisme de vote à deux étages : en analysant l'image dans différents espaces de représentation, un premier vote permet de déterminer les lieux de provenance les plus plausibles dans chacun de ces espaces séparément, avant de fusionner les notes obtenues dans un second étage de vote (voir figure 1.22). A la fin de cette procédure, si le niveau de confiance dans le vote final n'est pas satisfaisant, une nouvelle image est acquise. Si après avoir traité un certain nombre d'images ce seuil n'est toujours pas atteint, le superviseur indique le lieu actuel. Si le seuil de confiance est dépassé, le superviseur vérifie le lieu prédit par le robot afin de le corriger en cas d'erreur. Après chaque interaction avec le superviseur, le modèle de l'environnement est mis à jour : l'apprentissage est donc réalisé en-ligne, en mémorisant simplement les lieux dans lesquels chaque primitive visuelle a été vue. L'approche proposée ici présente de nombreuses qualités, notamment sur le traitement complètement incrémentiel (i.e., depuis la construction du modèle de l'environnement jusqu'à l'apprentissage des lieux correspondants). Cependant, elle repose sur une interaction régulière avec un superviseur, ce qui est en dehors du cadre que nous nous fixons dans les travaux rapportés dans ce mémoire.

## 1.4 Discussion

Dans cette section, nous proposons une discussion des différentes approches recensées ci-dessus en les regroupant en fonction de deux critères déterminant pour les distinguer : l'information visuelle prise en compte pour la caractérisation des images et de l'environnement, et la méthode de filtrage qu'elles emploient.

### 1.4.1 Information visuelle

Le but de cette section est de résumer les caractéristiques des méthodes recensées dans cet état de l'art en les présentant du point de vue de l'information visuelle sur laquelle elles se basent. Le choix de l'information visuelle est primordial, car celle-ci constitue le point d'entrée du processus d'estimation. On cherchera donc naturellement à optimiser le pouvoir expressif et discriminant de cette information. Cependant, les faiblesses de l'information visuelle au niveau de la discrimination peuvent dans certains cas être comblées par les méthodes d'estimation employées pour détecter les fermetures de boucle ou localiser le robot. C'est le cas notamment en présence d'aliasing perceptuel : lorsque plusieurs lieux distincts se ressemblent, l'information visuelle seule ne permet pas de prendre une décision. Dans ce genre de situation, un cadre d'estimation permettant de fusionner l'information au cours du temps permet de lever l'ambiguïté. Le caractère informatif de la représentation de l'image doit être considéré comme un compromis à faire avec

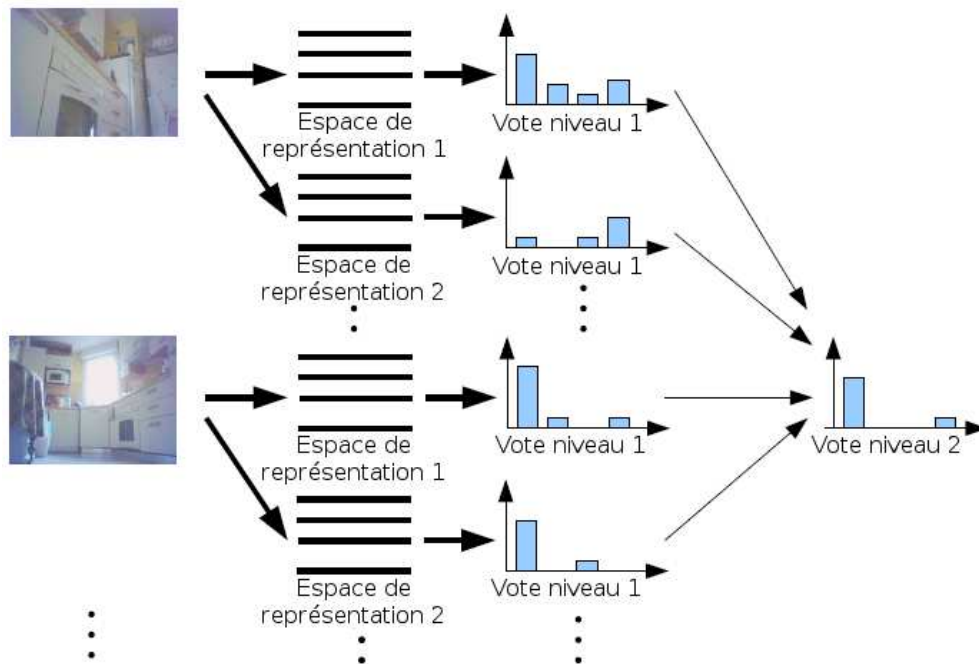


FIG. 1.22: Illustration du processus de vote à deux niveaux mis en oeuvre dans [Filliat, 2007]. Lorsqu'une nouvelle image est acquise, elle est caractérisée dans différents espaces de représentation qui votent tous à un premier niveau pour déterminer les lieux plausibles de provenance de cette image. Ces votes sont ensuite fusionnés à un second niveau pour obtenir la classe de l'image. Source : [Filliat, 2007].

le coût des traitements correspondants : plus on retient d'information, plus les traitements sont complexes et requièrent une puissance de calcul élevée. La méthode utilisée pour le traitement de l'image doit donc également prendre ce critère en compte, en cherchant au final à obtenir une représentation qui soit optimale du point de vue de la quantité d'information à la fois en termes d'expressivité, mais également en termes de coûts de traitements.

La plupart des méthodes présentées dans cet état de l'art reposent sur un seul type d'information visuelle, les images étant alors encodées dans un unique espace de représentation. La majorité de ces méthodes ([Andreasson et al., 2005], [Barfoot, 2005], [Booi et al., 2007], [Cummins and Newman, 2007], [Elinas et al., 2006], [Fraundorfer et al., 2007], [Ho and Newman, 2007], [Kosecká et al., 2005], [Newman et al., 2006], [Se et al., 2005], [Se et al., 2002], [Sim et al., 2005], [Sim and Dudek, 2004], [Tully et al., 2007], [Valgren et al., 2007], [Wang et al., 2006], [Zivkovic et al., 2005]) utilisent des primitives locales de type SIFT (*Scale Invariant Feature Transform*, [Lowe, 2004]), ou une version légèrement modifiée de ce genre de primitive (voir figure 1.23 pour une illustration de la construction du descripteur SIFT). Celles-ci présentent des qualités impressionnantes de robustesse aux changements d'orientation et d'échelle, mais également une

robustesse partielle aux changements d'illumination et aux variations affines de point de vue. Notamment, d'après les résultats des évaluations comparant différentes signatures d'images qui sont rapportés dans [Mikolajczyk and Schmid, 2003] et dans [Ramisa et al., 2008], les meilleures performances pour l'appariement ont été obtenues en utilisant les primitives SIFT. Cependant, un extracteur de primitives plus récent, *Speeded Up Robust Features* (SURF, [Bay et al., 2006]), semble plus efficace en coûts de traitements et quasiment aussi performant en termes de reconnaissance. Ce type de primitive a par ailleurs été employé pour la détection de fermeture de boucle [Cummins and Newman, 2008b] dans le cadre du SLAM topologique basé sur l'apparence uniquement. Les simples coins de Harris [Harris and Stephens, 1988] exhibent des performances de reconnaissance acceptables dans le cadre du suivi de points d'intérêt, et ils autorisent des calculs très rapides. En conséquence, ils sont toujours employés dans certaines applications ([Clemente et al., 2007], [Eustice et al., 2004], [Lemaire et al., 2007], [Williams et al., 2007b]) où le nombre d'images traitées par seconde est important (il s'agit généralement d'algorithmes de SLAM nécessitant des fréquences d'acquisition élevées). Il existe également d'autres primitives locales, conçues spécialement pour la localisation globale ou la détection de fermeture de boucle dans les travaux de [Dellaert et al., 1999a], [Frintrop and Cremers, 2007], [Karlsson et al., 2005] et [Wolf et al., 2005]. De même, certaines approches sont basées sur des signatures globales telles que les histogrammes de couleur ([Ulrich and Nourbakhsh, 2000]), CRFH (*Composed Receptive Field Histograms*, [Luo et al., 2007], [Pronobis et al., 2006]), les coefficients de Fourier ([Menegatti et al., 2004]), ou encore un simple vecteur en niveau de gris pris dans des images panoramiques ([Hubner and Mallot, 2007]). En définitive, il semble que des primitives locales avancées telles que SIFT et SURF constituent le type d'information visuelle préféré : non seulement ces primitives permettent des appariements robustes, même en cas d'importantes variations dans l'image, mais elles offrent aussi une plus grande quantité d'information et plus de souplesse dans la représentation que les signatures globales. Elles présentent de plus une robustesse accrue face aux occultations partielles dans l'image, même si elles sont plus sensibles au bruit local.

D'après les résultats des évaluations de [Mikolajczyk and Schmid, 2003], il apparaît que les performances d'appariements peuvent être améliorées de manière significative en combinant plusieurs représentations de la même image (i.e., dans différents espaces de caractérisation). C'est par exemple ce que proposent les auteurs de [Pronobis and Caputo, 2007], qui associent des primitives SIFT à la description globale donnée par les CRFH. De même, dans les travaux de [Filliat, 2007], les points d'intérêt SIFT sont utilisés en conjonction avec des histogrammes locaux de teinte, de manière à extraire à la fois une information de texture et de couleur dans les images. En conséquence, lorsque des images présentent une carence en texture (comme lorsque le robot est face à un mur uniformément coloré), les primitives SIFT sont difficilement extraites, et apportent peu d'information, alors que l'information de couleur est pertinente.

Afin d'améliorer les performances en termes de coûts de traitements, certains auteurs ([Goedemé et al., 2007], [Weiss et al., 2007]) proposent des approches hybrides pour la comparaison d'images, alternant entre représentations locales et globales en fonction des conditions. La similarité locale, qui offre des apparie-

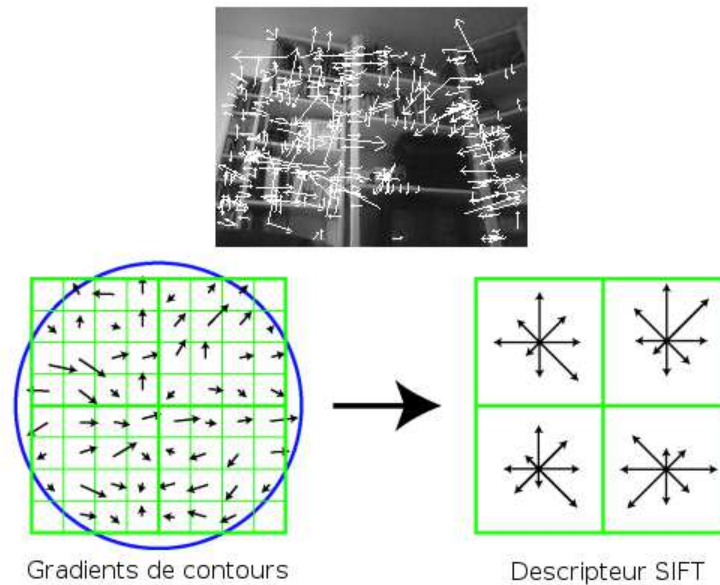


FIG. 1.23: Illustration du principe de construction du descripteur SIFT. Le descripteur est construit en deux étapes. Dans un premier temps, on calcule les orientations et les amplitudes des gradients pris aux alentours d'un point d'intérêt dans l'image (partie gauche de la figure). Ces informations sont alors pondérées par des coefficients Gaussiens (le cercle sert à délimiter la zone où ces coefficients sont non-nuls), avant d'être accumulées sous la forme d'histogrammes d'orientations regroupant l'information par sous-régions de 4x4 pixels. Le résultat de cette accumulation est illustré dans la partie droite de la figure, où la taille de chaque flèche dépend des amplitudes des gradients. Pour les besoins de la figure, seulement 4 sous-régions de 4x4 pixels chacune sont montrées, alors qu'en réalité 16 sous-régions de cette taille sont utilisées. Source : [Lowe, 2004].

ments plus précis mais plus coûteux en temps de calcul, n'est estimée que si la comparaison au niveau des signatures globales est satisfaisante ([Goedemé et al., 2007]) : cela permet d'éviter des traitements inutiles lorsque la comparaison globale a été infructueuse. Dans un contexte similaire, les auteurs de [Weiss et al., 2007] proposent de n'évaluer la similarité locale que dans les situations où l'incertitude liée au processus d'estimation est importante, requérant de ce fait une comparaison précise. Par ailleurs, dans [Filliat, 2007] et [Pronobis and Caputo, 2007], chaque image est graduellement traitée jusqu'à ce qu'une décision de localisation globale satisfaisante soit émise. Cela permet parfois d'éviter le traitement complet de l'image lorsqu'une information partielle est suffisante. Dans une perspective équivalente, les auteurs de [Fraundorfer et al., 2004], [Kim and Kweon, 2007] et [Lemaire et al., 2007] proposent de construire des constellations de simples points d'intérêt (i.e., des coins de Harris) afin d'améliorer les appariements tout en offrant des coûts acceptables pour les traitements de l'image. Dans [Fraundorfer et al., 2004], une ellipse est construite autour de groupes de points d'intérêt proches dans l'image puis normalisée au rayon unitaire, ce qui permet d'atteindre un niveau respectable d'invariance au changement de point de vue. Dans [Lemaire et al., 2007], des groupes de primitives voisines sont appariées entre deux images en alignant ces groupes par le

biais de simples rotations dans le plan. Dans [Kim and Kweon, 2007], des triplets de points d'intérêt générés aléatoirement servent à définir des cercles qui sont ensuite caractérisés grâce au descripteur SIFT. Les trois techniques qui viennent d'être mentionnées permettent d'atteindre des performances de reconnaissance proches de celles de SIFT, tout en autorisant des coûts de traitements beaucoup plus faibles. Enfin, les auteurs de [Tamimi and Zell, 2005] appliquent la méthode des "invariants d'intégrale" décrite dans [Wolf et al., 2002] à des primitives extraites selon la méthode SIFT : cela permet de se passer du calcul du descripteur SIFT et de le remplacer par une alternative presque aussi efficace mais plus simple à calculer.

L'importante quantité de types de primitives locales et globales qui existent dans la littérature et leur hétérogénéité montrent à quel point la conception d'une représentation optimale, à la fois en termes d'expressivité et de ressources nécessaires, est compliquée. Cela est généralement très fortement lié à la tâche pour laquelle cette représentation sera employée, mais cela dépend également du type d'environnement (i.e., intérieur ou extérieur). Plusieurs formalismes ont été proposés pour efficacement maintenir et gérer les caractérisations des images, mais aussi pour définir clairement les distances correspondantes pour les comparaisons. Par exemple, le paradigme des sacs de mots visuels ([Csurka et al., 2004], [Nilsback and Zisserman, 2006], [Nister and Stewenius, 2006]), que l'on utilise notamment dans les travaux rapportés dans ce mémoire, est bien adapté aux représentations basées sur tout type de primitive locale. Dans ce paradigme, une image est représentée par un ensemble de primitives élémentaires non-ordonné, les *mots*, choisis dans un *dictionnaire* (ou *vocabulaire*). La figure 1.24 donne une illustration de la caractérisation d'une image selon ce formalisme. Le dictionnaire est quant à lui construit par l'agglomération de descripteurs de primitives locales similaires. Généralement, cette phase de construction est réalisée hors-ligne, à partir d'images représentatives de l'environnement, même s'il est possible de s'affranchir de cette tâche de façon incrémentielle ([Filliat, 2007]). Encoder des images selon ce paradigme permet d'améliorer la robustesse au bruit local dans l'image, et cela offre l'avantage de s'adapter facilement à différents types de contextes, tels que l'apprentissage de modèle de l'apparence ([Cummins and Newman, 2007], [Cummins and Newman, 2008a]), la construction de matrice de similarité ([Ho and Newman, 2007], [Newman et al., 2006]), ou encore les méthodes de vote ([Filliat, 2007], [Fraundorfer et al., 2007], [Wang et al., 2006]).



FIG. 1.24: Représentation d'une image selon le paradigme des sacs de mots visuels. Les primitives extraites dans l'image courante sont mises en correspondance avec les mots du dictionnaire pour décrire l'image en fonction de l'occurrence de ces mots.



Une alternative aux sacs de mots visuels proposant un formalisme pour la représentation des images consiste à employer les méthodes d'Analyse en Composantes Principales (ACP), comme dans les travaux de [Gaspar et al., 2000], [Kröse et al., 2002] et [Sim and Dudek, 1999]. Les méthodes d'ACP permettent d'apprendre un espace optimisé pour la caractérisation des images, avec un nombre limité de dimensions, en ne retenant que celles qui contiennent le plus d'information. Les images sont par la suite projetées dans la base optimisée de cet espace pour en déterminer la similarité avec les images d'entraînement. Malgré des caractéristiques intéressantes concernant le gain en efficacité pour la comparaison des images, les techniques d'ACP mentionnées ici requièrent une phase préalable hors-ligne pour apprendre le modèle de l'environnement (i.e., l'espace optimisé de projection). Elles dépendent par ailleurs fortement des images d'entraînement utilisées pour cet apprentissage, contraignant alors par la suite les déplacements du robot dans l'environnement appris uniquement.

Enfin, il faut également mentionner les Randomized Tree ([Lepetit and Fua, 2006]) qui permettent de considérer la reconnaissance de primitives comme une tâche de classification. Cette approche repose sur la construction d'une forêt d'arbres de décisions binaires pour la classification des primitives. Chaque arbre, entraîné au préalable sur la base d'exemples d'apprentissage, est traversé par une suite d'opérations rapides (i.e., de simples comparaisons de pixels), ce qui permet d'atteindre des performances en temps réel même à des fréquences relativement élevées (i.e., 30Hz) pour l'acquisition des images. Les auteurs de [Williams et al., 2007a] proposent notamment une version incrémentielle des RT pour le recalage de la position de la caméra suite à un kidnapping, ou encore pour la détection de fermeture de boucle [Williams et al., 2008] dans le cadre d'un algorithme de SLAM visuel ([Davison et al., 2007]).

### 1.4.2 Les processus d'inférence

Dans la dernière sous-section de cet état de l'art, nous présentons les différents formalismes proposés dans les méthodes détaillées jusque-là pour effectuer l'estimation du lieu de provenance de l'image courante. Il a déjà été souligné l'importance de reposer sur un cadre robuste pour l'inférence du lieu de chaque image. Cela permet en particulier de gérer des situations ambiguës et, lorsque cela s'avère plus sûr, de s'abstenir d'une prise de décision afin d'éviter une erreur. La principale caractéristique de ces formalismes d'inférence est de pouvoir fusionner les estimés au cours du temps, afin de prendre en compte l'information des états précédent lors de l'estimation courante. Il existe par ailleurs des formalismes [Goedemé et al., 2007] qui proposent de modéliser l'ignorance et l'absence d'information, reposant pour cela sur la théorie de l'évidence.

On peut remarquer que parmi les méthodes présentées ici pour la détection de fermeture de boucle, ce sont les méthodes basant leur critère d'estimation sur le maximum de vraisemblance (MDV) qui offrent les implémentations les plus simples. En effet, comme le montrent les travaux de [Booi et al., 2007], [Eustice et al., 2004], [Fraundorfer et al., 2007], [Hubner and Mallot, 2007], [Kosecká et al., 2005], [Lemaire et al., 2007], [Se et al., 2002], [Ulrich and Nourbakhsh, 2000], [Wang et al., 2006] et [Williams et al., 2007b],

les méthodes de vote peuvent être mises en oeuvre avec succès, conduisant à des traitements simples de l'information visuelle et à une gestion aisée des hypothèses. Généralement, les résultats du vote ont besoin d'être confirmés (employant pour cela une évaluation qualitative [Ulrich and Nourbakhsh, 2000] ou un algorithme de géométrie multi-vues [Booi et al., 2007], [Eustice et al., 2004], [Fraundorfer et al., 2007], [Kosecká et al., 2005], [Se et al., 2002], [Wang et al., 2006], [Williams et al., 2007b]) dans le but d'améliorer la robustesse du processus d'inférence en écartant les données aberrantes.

Toutefois, les approches basées sur le critère du MDV souffrent d'un certain nombre de limitations. Tout d'abord, elles reposent généralement sur des comparaisons exhaustives de l'image courante avec l'ensemble des entités du modèle de l'environnement pour en déduire les hypothèses les plus vraisemblables : le processus résultant nécessite donc d'importantes ressources, surtout dans des environnements de grande taille. Deuxièmement, ce genre de technique n'est pas adapté aux situations où plusieurs hypothèses coexistent au cours du temps, étant donné qu'elles ne peuvent pas discriminer clairement entre les hypothèses vraisemblables. Ainsi, il arrive dans ces cas qu'une décision erronée soit prise, ce qui est fréquent en présence d'aliasing perceptuel. Enfin, le critère du MDV est sujet aux erreurs temporaires de détection (i.e., lorsque la ressemblance avec une entité du modèle n'est dû qu'à un phénomène passager dans l'image courante, qui ne durera pas au cours du temps). Une illustration de ce phénomène et de la faiblesse du critère du MDV dans ce cas est donné dans la figure 1.2.

Pour palier ces limitations, plusieurs approches ont été proposées, afin d'assurer la viabilité de l'association de données avant d'entériner définitivement une hypothèse. Par exemple, l'algorithme GCBB [Neira et al., 2003] employé dans les travaux de [Clemente et al., 2007] permet de n'accepter une hypothèse de fermeture de boucle que si les primitives locales avoisinant la position actuellement estimée sont appariées de manière cohérente avec des amers de la carte selon le *Joint Compatibility Test* (JCT, [Neira and Tardós, 2001]). Dans une perspective similaire, cette viabilité peut être assurée en sélectionnant les hypothèses de fermeture de boucle comme étant les éléments hors-diagonaux d'une matrice de similarité ([Ho and Newman, 2007], [Newman et al., 2006]) : on impose de cette façon la cohérence temporelle de la détection. Cependant, cela nécessite des comparaisons exhaustives et coûteuses entre l'image courante et le modèle de l'environnement, ainsi que des manipulations de matrice à la complexité cubique en le nombre d'éléments qu'elle contient. Bien qu'elle soit incrémentielle, la technique *incremental spectral clustering* développée par les auteurs de [Valgren et al., 2007] requière elle aussi des manipulations complexes sur des matrices. Une approche plus prometteuse est décrite dans les travaux de [Cummins and Newman, 2007], où un modèle génératif de l'apparence de l'environnement est appris, permettant une estimation de la probabilité de fermeture de boucle au sens du maximum a posteriori (MAP). La complexité résultante est linéaire en le nombre de lieux du modèle, et il est également possible d'optimiser le processus d'estimation [Cummins and Newman, 2008b] pour en rendre les traitements plus efficaces. En dépit de ces améliorations, l'estimation n'est toutefois pas effectuée en temps réel. D'autre part, l'approche repose toujours sur une phase préalable hors-ligne d'apprentissage du modèle.

Il est aussi possible d'apprendre des modèles génératifs de l'apparence de l'environnement produisant une estimation métrique de la position, comme le montrent les auteurs de [Kröse et al., 2002], [Sim and Dudek, 1999] et [Sim and Dudek, 2004]. Cependant, la phase d'apprentissage sous-jacente est très lourde, nécessitant en particulier le traitement d'un large quantité d'images d'entraînement prises à partir de points de vue proches. En conséquence, il paraît difficile d'utiliser ce genre de modèle dans le cas d'environnement de grande taille.

D'autres formalismes d'inférence, tels que les classeurs du type MSV (Machines à Support Vecteur), ont été investigués dans les travaux de [Luo et al., 2007], [Pronobis and Caputo, 2007] et [Pronobis et al., 2006]. Ceux-ci offrent notamment des capacités d'adaptation aux changements de condition dans l'environnement ([Luo et al., 2007]), et la complexité des traitements qu'ils requièrent peut être gérée pour croître de manière progressive au fur et à mesure du traitement d'une image ([Pronobis and Caputo, 2007]). Toutefois, les techniques à base de MSV ne sont pas adaptées aux problèmes avec un nombre important de classes (puisque dans ce cas l'apprentissage d'un modèle devient une tâche lourde), et un nombre faible d'exemples d'entraînement pour chaque classe. Il semble donc que ce genre d'approche soit utile pour la reconnaissance globale de lieux, lorsque le but consiste à discriminer parmi quelques lieux distincts, mais probablement pas à la détection de fermeture de boucle, où une petite portion de la trajectoire passée de la caméra doit être reconnue à partir d'un nombre limité d'images.

Enfin, les approches d'échantillonnage approximant la distribution de probabilité de la position du robot ([Andreasson et al., 2005], [Dellaert et al., 1999a], [Menegatti et al., 2004], [Weiss et al., 2007], [Wolf et al., 2005]) ont été implémentées avec succès dans le cadre de la localisation globale et du SLAM métriques. Les méthodes sous-jacentes sont spécialement conçues pour approximer des probabilités de distribution non-paramétriques, à la forme inconnue, dans un espace d'état continu, à partir d'un ensemble fini d'échantillons discrets. Cependant, pour la détection de fermeture de boucle, lorsque l'environnement est modélisé par une représentation topologique discrète (comme c'est le cas dans les approches de reconnaissance et de classification d'image recensées ici), une probabilité de distribution sur cet espace d'état peut être maintenue et gérée plus facilement, avec un filtre Bayésien discret par exemple.

## 1.5 Conclusion sur l'état de l'art

La densité et la diversité des méthodes (et des modèles sous-jacents) recensées dans cet état de l'art montrent à quel point le problème de l'association de données pour la détection de fermeture de boucle et la localisation globale est difficile. Il semble en effet qu'il n'y ait pas de solution générale applicable dans tous les cas, tant ce problème semble avoir été abordé de différentes manières. On trouve par exemple des extensions à des algorithmes de SLAM, basant leur technique sur un modèle de l'environnement qui n'est pas adapté à la tâche abordée, et qui est contraint en qualité et en quantité par le processus de SLAM : cela interdit toute adaptation directe à des algorithmes de SLAM reposant des modèles de l'environnement dif-

férents. D'autres approches requièrent une phase d'apprentissage préalable hors-ligne, étape contraignante au cours de laquelle une quantité importante d'images représentatives de l'environnement (certaines fois localisées grâce à une vérité terrain) doivent être analysées et encodées pour en obtenir un modèle pertinent de l'environnement. Le caractère hors-ligne mis à part, ces approches ont souvent le défaut de limiter les déplacements du robot dans les zones correctement apprises.

En définitive, il semble qu'aucune des approches citées plus haut ne permette de satisfaire à la fois toutes les contraintes fixées dans l'introduction de ce mémoire. Les méthodes temps réel reposent souvent sur un modèle bâti dans le cadre d'un processus de SLAM, alors que les techniques construisant leur propre modèle de l'environnement dépendent d'une phase d'apprentissage hors-ligne, ou bien nécessitent des traitements dont la complexité laisse difficilement envisager une réalisation en temps réel.

Le tableau 1.1 récapitule les caractéristiques de certaines des approches citées dans cet état de l'art. Les caractéristiques retenues sont la nécessité d'une phase d'apprentissage préalable hors-ligne (oui / non), la capacité à réaliser la détection de fermeture de boucle en temps réel (oui / non), la construction d'un modèle de l'environnement indépendant de tout algorithme de SLAM (oui / non), l'indépendance vis-à-vis d'une estimation de la position pour la détection de fermeture de boucle (oui / non), et enfin la robustesse à l'aliasing perceptuel (— / — / + / + +). N'apparaissent dans ce tableau qu'un nombre restreint de méthodes, afin d'en faciliter la lisibilité. Nous avons d'une part sélectionné les formalismes généraux qui ont été implémentés par différents auteurs, comme le MCL ou le RBpf. Nous avons d'autre part choisi les approches qui nous ont semblé les plus pertinentes pour résoudre le problème de la détection de fermeture de boucle, au sens des critères énoncés en introduction (i.e., [Newman et al., 2006], [Cummins and Newman, 2008b]). Enfin, nous avons fait apparaître des travaux récents qui proposent des solutions efficaces pour améliorer les algorithmes de SLAM visuel existants ([Clemente et al., 2007], [Williams et al., 2007a]), puisque c'est le but visé ici.

TAB. 1.1: Récapitulatif.

Méthode	Apprentissage HL	DFB Tps réel	Modèle indép.	Dépend position	Robustesse AP
[Williams et al., 2007a]	non	oui	non	non	—
[Clemente et al., 2007]	non	non	non	oui	+ (JCT)
MCL / RBpf	non	oui	non	non	+
[Newman et al., 2006]	oui	non	oui	non	++
[Cummins and Newman, 2008b]	oui	non	oui	non	++

En dépit du constat dressé ci-dessus, les travaux de [Cummins and Newman, 2008b] se rapprochent fortement du cadre défini par les contraintes énoncées en introduction, avec un modèle considérant la détection de fermeture de boucle comme une tâche de classification d'image, et avec une complexité raisonnable lors de l'exploitation. L'approche proposée nécessite toutefois une phase d'apprentissage hors-ligne du modèle de l'environnement, mais celui-ci peut être utilisé par la suite dans différents contextes (i.e., le robot n'est pas strictement restreint au domaine appris). L'approche présentée dans ce mémoire partage donc un

certain nombre de points communs avec cette méthode, mais elle permet également la réalisation des traitements dans un cadre complètement incrémentiel (i.e., de l'apprentissage d'un modèle de représentation des images jusqu'à la détermination du lieu de l'image courante), sans nécessiter de phase préalable hors-ligne d'apprentissage, tout en offrant des performances en temps réel pour la détection de fermeture de boucle.



## Chapitre 2

# Cadre Bayésien pour la détection de fermeture de boucle

Dans ce mémoire, nous proposons d’aborder le problème de la détection de fermeture de boucle du point de vue d’une tâche de *classification* d’images : lorsqu’une image est acquise, il faut en déterminer la *classe*, c’est à dire le lieu auquel elle appartient. S’il s’avérait que cette image vienne d’un lieu connu, cela signifierait que le robot est retourné sur ses pas et qu’une boucle vient d’être fermée. Pour déterminer la classe de chaque image perçue, un modèle de l’environnement doit être construit, celui-ci renseignant sur la description de chacun des lieux visités jusque-là. Ce modèle, initialement vide, doit être mis à jour de manière *incrémentielle* au fur et à mesure que le robot évolue dans son environnement : chaque fois qu’une nouvelle image est acquise, il faut déterminer si elle provient d’un lieu déjà visité (et dont une représentation existe dans le modèle établi jusque-là) ou bien si elle caractérise un nouveau lieu. Ainsi, c’est le résultat de l’opération de classification de l’image courante qui permet de déterminer les mises à jour à effectuer sur le modèle. Comme le précisent les auteurs de [Ranganathan et al., 2006], lorsqu’une telle représentation de l’environnement est inférée à partir des observations faites par le robot, le modèle obtenu est une partition de l’ensemble des perceptions. En conséquence, si  $M_t = \{L_i\}_{i=0}^m$  constitue l’ensemble des lieux mémorisés à l’instant  $t$  dans le modèle de l’environnement à partir de la séquence  $I^t = I_0, \dots, I_t$  d’images acquises jusque-là, alors  $m \leq t$ .

Le processus global conduisant à l’inférence d’un modèle cohérent de l’environnement ainsi que nous venons de le décrire est schématisé par le diagramme de la figure 2.1. Lorsqu’une nouvelle image est acquise, elle est tout d’abord encodée sous la forme d’un ensemble de primitives locales, selon une méthode incrémentielle de *sacs de mots visuels* [Filliat, 2007]. Ensuite, elle est comparée à la dernière image traitée dans le cadre de ce processus afin d’éviter les *détections locales de fermeture de boucle* (phénomène dû à la nature consécutive de l’acquisition des images et particulièrement remarquable lorsque la caméra est immobile et que les images sont toutes semblables). Si l’image n’a pas été écartée, elle est prise en compte

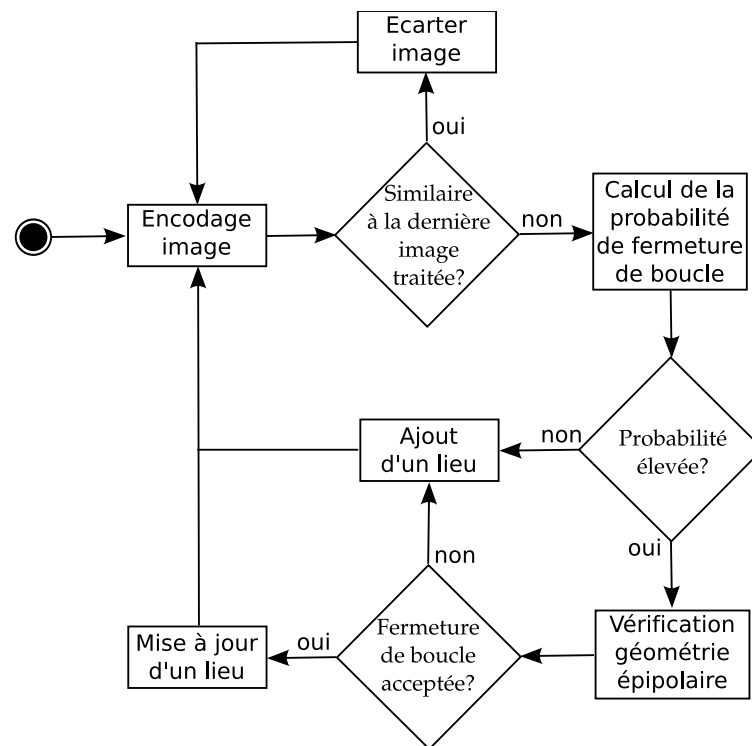


FIG. 2.1: Diagramme du processus global de la détection de fermeture de boucle (voir le texte pour les détails).

pour l'estimation de la probabilité de fermeture de boucle, dans un cadre de filtrage Bayésien. Lorsque cette probabilité est élevée, un algorithme de géométrie multi-vues [Nistér, 2004] est employé pour vérifier que la contrainte de géométrie épipolaire [Hartley and Zisserman, 2004] est satisfaite (voir figure 2). Ce test permet d'assurer l'existence d'une structure commune entre l'image considérée et le lieu présumé de provenance avant de valider définitivement la fermeture de boucle : il peut arriver en effet que l'image courante ressemble fortement à un lieu du modèle sans pour autant en provenir (on parlera alors d'*aliasing perceptuel*, voir section 1.1.2 de l'état de l'art de cette partie), auquel cas la vérification précise de la structure sous-jacente permet de lever l'ambiguïté. Au final, si une fermeture de boucle a été détectée, le lieu auquel appartient l'image courante est mis à jour avec la description de cette dernière. Le cas échéant, cette description servira à la création d'un nouveau lieu dans le modèle. Les différentes étapes de ce processus global sont détaillées dans les différentes sections de ce chapitre.



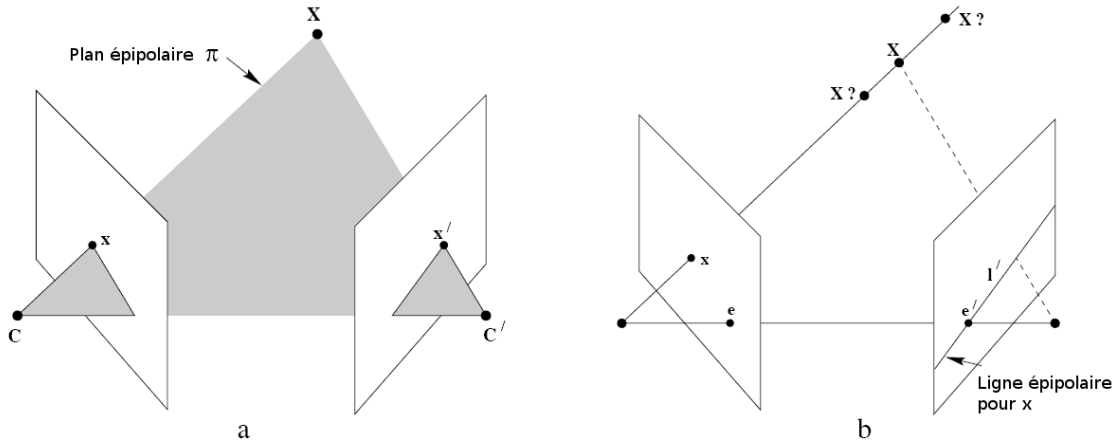


FIG. 2.2: Géométrie épipolaire. (a) Les deux caméras sont représentées par leurs centres de projection  $C$  et  $C'$  et les plans image correspondants. Ces centres de projection, le point 3D  $X$  et ses projections  $x$  et  $x'$  dans les deux plans image appartiennent à un même plan  $\pi$ . (b) La profondeur du point image  $x$  peut être inférée sur la demi-droite de l'espace 3D partant de  $C$  et passant par  $x$ . La projection de cette demi-droite intersecte le plan image de la seconde caméra pour former la ligne  $l'$ . En conséquence, l'image  $x'$  de  $X$  dans la deuxième vue appartient à  $l'$ . Vérifier la contrainte de géométrie épipolaire revient à s'assurer que les images  $x$  et  $x'$  d'un point  $X$  dans deux vues distinctes appartiennent à un même plan défini par les centres de projection  $C$  et  $C'$  de ces deux vues, et le point  $X$ . Source : [Hartley and Zisserman, 2004].

## 2.1 Représentation des images et définition des lieux du modèle

Afin d'assurer l'efficacité et la robustesse de la détection de fermeture de boucles, chaque lieu du modèle de l'environnement doit être caractérisé de façon compacte et expressive : il ne faut retenir que l'information pertinente. D'autre part, étant donné qu'un même lieu peut-être décrit par plusieurs images (i.e., en cas de multiples passages au même endroit), la représentation sous-jacente doit être extensible pour pouvoir être mise à jour si besoin est. Pour autant, il ne serait pas judicieux de caractériser un lieu directement par les images qui lui appartiennent, étant donné que la majeure partie de l'information contenue dans ces images est inutile.

Dans les travaux présentés dans ce mémoire, nous avons choisi de représenter les images selon l'approche des sacs de mots visuels. En conséquence, chaque lieu du modèle de l'environnement sera décrit par la collection des mots visuels trouvés dans les images caractérisant ce lieu. Ce choix a été principalement motivé par deux raisons. Tout d'abord, comme remarqué dans l'état de l'art (chapitre 1) de cette partie du mémoire, les représentations basées sur les primitives locales apportent plus de souplesse à la caractérisation de l'image que les représentations globales : elles sont notamment plus adaptées en cas d'occultations partielles. De plus, le cadre fonctionnel proposé par la méthode des sacs de mots visuels permet, comme nous le verrons par la suite, de s'accommoder simplement et uniformément de tout type de primitive locale

afin de mixer diverses sources d'information (la texture et la couleur par exemple) et les représentations correspondantes.

### 2.1.1 Sacs de mots visuels

La méthode sacs de mots visuels est une approche populaire dans le cadre de la catégorisation d'images ([Csurka et al., 2004], [Nilsback and Zisserman, 2006], [Nister and Stewenius, 2006]). Elle repose sur une représentation des images sous la forme d'un ensemble non-ordonné de primitives locales, les *mots*, choisies à partir d'un *dictionnaire* (ou *vocabulaire*). Généralement, le dictionnaire est appris lors d'une phase préalable hors-ligne, à partir d'images représentatives pour la tâche à accomplir. La construction du dictionnaire consiste en une clusterisation (selon la méthode des *k-means* par exemple) des descripteurs visuels associés aux primitives extraites dans des images d'entraînement. Cela permet notamment d'améliorer la robustesse au bruit local dans l'image lors de l'appariement ultérieur des primitives. Une fois un dictionnaire appris, on peut inférer la classe d'une image simplement sur la base de la fréquence des mots qu'elle contient, ignorant ainsi toute structure globale et améliorant de fait la robustesse aux occultations partielles. Les étapes de construction hors-ligne et d'utilisation du dictionnaire pour encoder les images sont illustrées dans la figure 2.3.

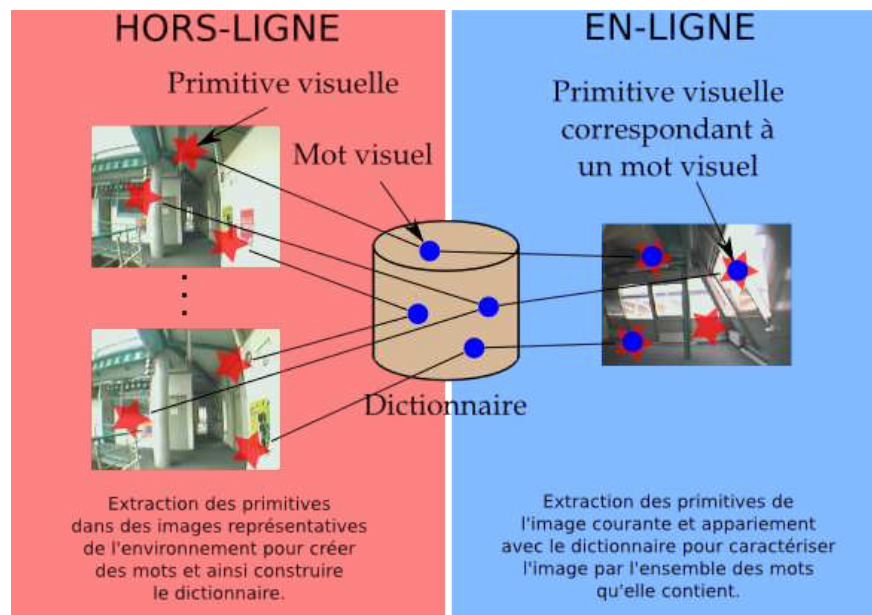


FIG. 2.3: Construction hors-ligne et utilisation du dictionnaire. Le dictionnaire est construit lors d'une phase préalable hors-ligne par agrégation de primitives visuelles extraites dans des images d'entraînement. Une fois la construction achevée, chaque image traitée est caractérisée par l'occurrence des mots trouvés dans cette image.

Dans le cadre des travaux présentés ici, nous avons choisi une variante incrémentielle [Filliat, 2007] de

la méthode des sacs de mots visuels : au lieu d'apprendre le vocabulaire au cours d'une phase préalable hors-ligne, comme c'est le cas dans la majorité des applications ([Cummins and Newman, 2007], [Cummins and Newman, 2008a], [Fraundorfer et al., 2007], [Ho and Newman, 2007], [Newman et al., 2006], [Wang et al., 2006]), cet apprentissage est effectué en-ligne, à partir d'une structure initialement vide qui est graduellement remplie au fil de la découverte de l'environnement. Pour cela, lorsqu'une image est traitée, les primitives visuelles qui n'ont pas d'équivalent dans le dictionnaire sont ajoutées à celui-ci comme de nouveaux mots (voir figure 2.4).

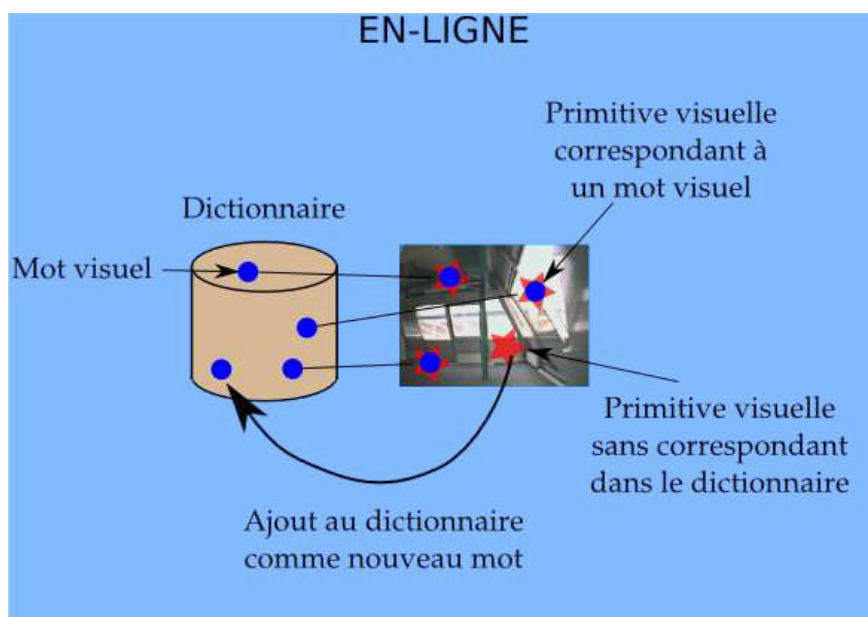


FIG. 2.4: Construction en-ligne et utilisation du dictionnaire. Le dictionnaire est construit en-ligne au fur et à mesure de la découverte de l'environnement : chaque primitive extraite dans l'image courante qui ne trouve pas d'équivalent dans le dictionnaire (i.e., qui ne correspond à aucun mot) est ajoutée à celui-ci comme nouveau mot.

Le mécanisme d'ajout d'un mot dans le dictionnaire repose sur le calcul de la distance entre les descripteurs de la primitive visuelle considérée et tous les mots du dictionnaire. On considère pour cela les mots du dictionnaire comme des sphères de rayon fixe dans l'espace des descripteurs. Ainsi, si la distance d'une primitive au centre d'une sphère est inférieure au rayon de cette sphère, la primitive est appariée au mot correspondant. Si la primitive ne tombe dans aucun des mots du dictionnaire, un nouveau mot est ajouté dans le dictionnaire en créant une nouvelle sphère centrée sur la primitive dans l'espace des descripteurs. La procédure d'ajout d'un mot dans le dictionnaire est détaillée dans la figure 2.5.

Grâce à la construction incrémentielle du vocabulaire, le système ne fait aucune hypothèse préalable sur le type d'environnement (i.e., intérieur ou extérieur) dans lequel le robot va évoluer, offrant ainsi une meilleure capacité d'adaptation à tout type d'environnement. Par ailleurs, une structure arborescente [Filliat,

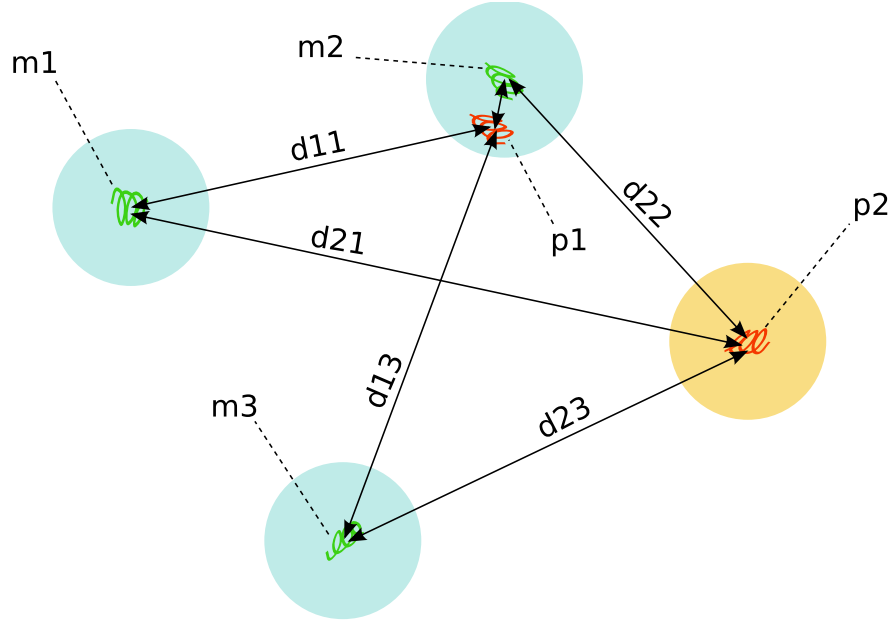


FIG. 2.5: Mécanisme d'ajout d'un mot dans le dictionnaire. La distance entre les descripteurs de la primitive  $p1$  et du mot  $m2$  est inférieure au rayon de la sphère enveloppant  $m2$  :  $p1$  est apparié avec  $m2$ . Par ailleurs, les distances  $d21$ ,  $d22$ ,  $d23$  entre les descripteurs de la primitive  $p2$  et des mots  $m1$ ,  $m2$  et  $m3$  sont toutes supérieures au rayon des mots :  $p2$  est ajoutée comme nouveau mot dans le dictionnaire.

2008], inspirée des travaux présentés dans [Nister and Stewenius, 2006], est utilisée ici pour améliorer les performances computationnelles lors de la recherche de mots correspondant à une primitive en particulier dans le dictionnaire. Cette structure est indispensable pour que tous les traitements décrits dans la figure 2.1 puissent être réalisés en temps réel, comme en attestent les résultats obtenus dans nos travaux précédents ([Angeli et al., 2008a], [Angeli et al., 2008b]), ainsi que dans le chapitre 3 de cette partie. Pour plus de détails sur l'implémentation incrémentielle de la méthode des sacs de mots visuels ou sur la structure arborescente, le lecteur intéressé pourra se référer à [Filliat, 2007] et à [Filliat, 2008], où l'auteur présente une évaluation de l'influence des paramètres régissant les algorithmes correspondant.

### 2.1.2 Espaces de représentation

Dans l'état de l'art de cette partie du mémoire (voir chapitre 1), nous avons souligné l'intérêt d'utiliser plusieurs caractérisations de la même image (i.e., dans différents espaces de représentation), afin d'en extraire des informations complémentaires. Comme le montrent les résultats obtenus dans [Filliat, 2007] et dans [Angeli et al., 2008b], cela permet d'améliorer les performances de classification d'image, généralisant les capacités obtenues avec un seul type de représentation à un ensemble plus varié d'environnements. Ainsi, dans les expériences rapportées ici, nous pourrions encoder les images de deux manières différentes. Dans le

premier cas, seulement une information locale de structure sera utilisée, construisant un dictionnaire sur la base de primitives SIFT ([Lowe, 2004]) extraites dans les images. Dans le second cas, une autre représentation sera employée en complément, construisant un dictionnaire d'histogrammes de teinte (i.e., histogramme des composantes H dans l'espace des couleurs HSV) en plus du vocabulaire des primitives SIFT.

Nous avons déjà évoqué précédemment les atouts des primitives SIFT (i.e., robustesse aux changements d'échelle et aux transformations géométriques du plan, tolérance acceptable aux changements de points de vue 3D et aux variations affines d'illumination), justifiant leur choix pour la tâche de classification d'images. Cependant, dans des environnements faiblement structurés, les images traitées n'exhibent pas assez de texture pour que le descripteur des primitives SIFT soit pertinent (voir l'image de la figure 2.6). Dans ce genre de situation, fréquente dans certains environnements d'intérieur aux murs uniformément colorés, l'adjonction d'un autre type de représentation, encodant une information de couleur, est approprié.



FIG. 2.6: Défaillance du descripteur SIFT : dans cette image prise dans un environnement d'intérieur, seulement 4 primitives SIFT ont pu être extraites en raison de la faible structure observée. Par ailleurs, la coloration des murs (teinte beige) semble être une information plus pertinente.

Voici un résumé des caractéristiques des différentes primitives et de la façon dont elles sont extraites (voir également la figure 2.7 pour un aperçu de la caractérisation des images dans les différents espaces de représentation) :

- SIFT [Lowe, 2004]

Les points d'intérêt sont détectés comme des extrema locaux dans l'espace et dans les différents niveaux d'échelle de différences de convolutions Gaussiennes obtenues à partir de l'image. Chaque primitive est caractérisée par un descripteur de dimension 128 qui est un histogramme des orientations des gradients calculés dans un voisinage du point considéré, à la position et à l'échelle correspondantes. Lors de l'appariement, les descripteurs sont comparés avec la norme L2.

- Histogrammes locaux de teinte

L'image est décomposée sous la forme d'un ensemble régulier de fenêtres de taille 40x40 pixels prises tous les 20 pixels. Les histogrammes de H normalisés dans l'espace des couleurs HSV pour chaque

fenêtre constituent les primitives de cette espace de représentation. Les descripteurs obtenus sont de dimension 16 et lors de l'appariement, ils sont comparés avec la distance de diffusion [Ling and Okada, 2006].

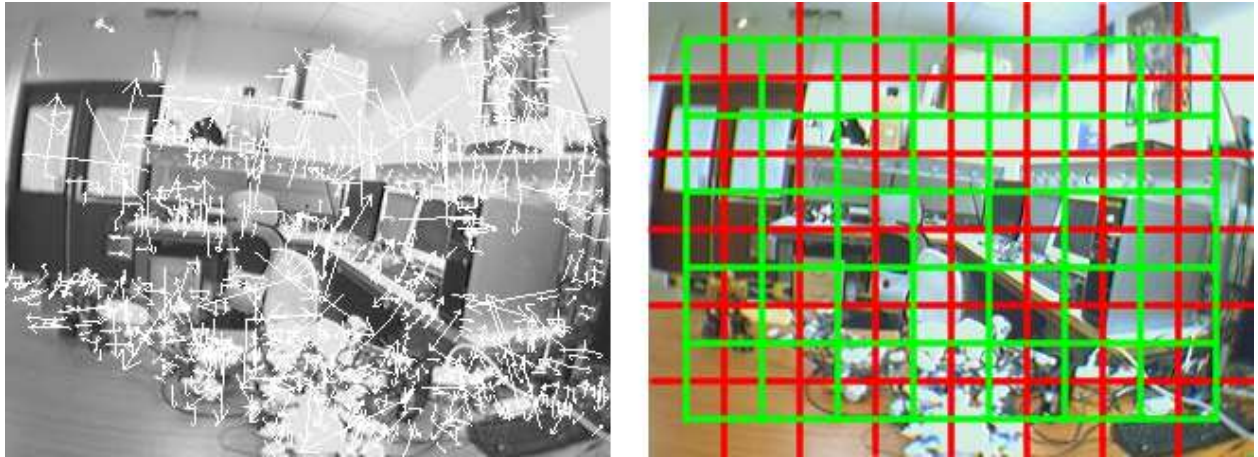


FIG. 2.7: *Caractérisation des images. Sur une image provenant d'un environnement d'intérieur sont superposés les descripteurs SIFT qui y ont été extraits (partie gauche de la figure) et les fenêtres dans lesquelles les histogrammes de teinte sont calculés (partie droite de la figure). En raison du chevauchement des fenêtres lors du calcul des histogrammes, deux couleurs distinctes ont été employées ici pour le quadrillage.*

Pour récapituler, les images sont encodées selon la méthode incrémentielle des sacs de mots visuels ([Filliat, 2007]), comme un ensemble de primitives locales prises dans différents espaces de représentation (i.e., SIFT et histogrammes H). En conséquence, chaque lieu du modèle de l'environnement est caractérisé par la collection des mots décrivant les images qui lui appartiennent, et ce dans tous les espaces de représentation pris en compte (voir figure 2.8).

## 2.2 Stratégie de sélection des images

Dans le cadre de notre solution de détection de fermeture de boucle, nous souhaitons associer une image à un lieu (existant ou non) de l'environnement. Pour cela, nous reposons sur les images fournies par une caméra monoculaire. Il n'est toutefois pas indispensable de traiter la totalité de ces images. En effet, en raison de la fréquence élevée de l'acquisition (i.e., entre 25 et 30 images par seconde avec les caméras employées ici), et en considérant une vitesse de déplacement raisonnable (i.e., quelques mètres par seconde), deux images consécutives proviennent de points de vue très proches.

Afin de ne pas sélectionner les images dont le point de vue est trop proche de celui de la dernière image traitée, on peut simplement imposer que la distance entre ces points de vue soit supérieure à un certain seuil. Cela implique néanmoins de pouvoir évaluer cette distance, à partir des mesures d'odométrie fournies



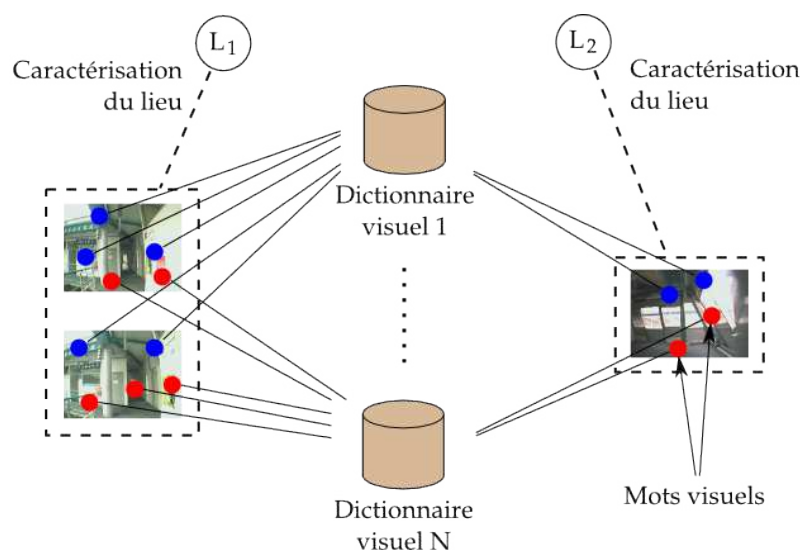


FIG. 2.8: Illustration de la caractérisation d'un lieu : un lieu est caractérisé par la collection des mots visuels trouvés dans les images appartenant à ce lieu. Ici, comme le lieu 1 dans la figure est un lieu de fermeture de boucle, il est décrit par les mots visuels provenant de 2 images différentes.

par le robot par exemple. Dans notre cas, nous ne disposons pas d'une telle information. Une approche naïve consisterait alors à échantillonner les images avec une fréquence plus basse : en augmentant le temps séparant deux images consécutives, on augmente les chances que la caméra se soit suffisamment déplacée. Ainsi, dans les expériences rapportées dans la suite de ce mémoire, les images sont acquises à 0.5Hz ou 1Hz, en fonction des caractéristiques de l'environnement (i.e., la profondeur de champ des scènes perçues notamment).

La stratégie naïve de sélection des images décrite ci-dessus présente toutefois un inconvénient majeur : lorsque la caméra est immobile, les images acquises sont toutes quasiment identiques : il n'est donc pas nécessaire de considérer l'intégralité de ces images, et seule la première d'entre elles (i.e., acquise au moment où la caméra s'est immobilisée) est utile, les autres pouvant être écartées. Dans ce but, nous avons définis une *mesure de similarité* permettant de comparer l'image courante  $I_t$  à un lieu  $L_i$  du modèle : il s'agit du pourcentage de primitives extraites dans l'image  $I_t$  qui sont des mots caractérisant le lieu  $L_i$ . Plus ce pourcentage est élevé, plus la similarité entre l'image et le lieu considérés est importante. Sur la base de ce critère, chaque image nouvellement acquise est sélectionnée pour le reste des traitements seulement si la mesure de similarité avec le dernier lieu ajouté ou mis à jour dans le modèle est inférieure à 90%. Cela permet d'écarter les images consécutives qui sont très similaires, et ainsi d'éviter des traitements inutiles.

### 2.3 Probabilité de fermeture de boucle

Afin de prédire correctement la classe d’une image, l’approche présentée ici définit un cadre probabiliste de filtrage Bayésien dont la fonction est de déterminer la probabilité que l’image courante vienne d’un des lieux déjà visités par le passé : c’est la probabilité de fermeture de boucle. On cherche donc, s’il existe, le lieu  $L_{i^*}$  du modèle  $M_{t-1}$  qui maximise cette probabilité :

$$L_{i^*} = \operatorname{argmax}_{i=-1, \dots, m} p(S_t = i | I_t, M_{t-1}) \quad (2.1)$$

où  $S_t = i$  est l’évènement “fermeture de boucle avec le lieu  $L_i$  à l’instant  $t$ ” (i.e., l’image courante  $I_t$  appartient à  $L_i$ ). L’évènement  $S_t = -1$  correspond à l’évènement “pas de fermeture de boucle à l’instant  $t$ ”, pour les cas où l’image courante ne vient pas d’un lieu existant déjà. Pour résoudre l’équation 2.1, il faut estimer la *probabilité a posteriori*  $p(S_t = i | I_t, M_{t-1})$  pour tout  $i = -1, \dots, m$ . D’après la loi de Bayes, cette probabilité peut être décomposée comme suit :

$$p(S_t | I_t, M_{t-1}) = \eta p(I_t | S_t, M_{t-1}) p(S_t | M_{t-1}) \quad (2.2)$$

où  $\eta$  est un terme de normalisation. Notons ici que lors du calcul de la probabilité a posteriori comme indiqué dans l’équation 2.2, l’estimation repose sur le modèle construit jusqu’au pas de temps  $t - 1$  (i.e.,  $M_{t-1}$ ), celui-ci n’étant pas ici une quantité estimée : la mise à jour du modèle, pour obtenir  $M_t$ , est en effet réalisée une fois que la probabilité a posteriori a été estimée, en fonction de sa distribution (voir figure 2.1).

Afin de prendre en compte dans l’équation 2.2 la caractérisation de l’image courante sous la forme des sacs de mots visuels, comme proposé dans la section 2.1.1, nous avons besoin d’introduire de nouvelles notations. Ainsi, soit  $(Z_k)_i$  l’état du dictionnaire associé à l’espace de représentation  $k$  (primitives SIFT ou histogrammes H dans ce mémoire) à l’instant  $i$ . L’indice  $i$  dans la notation  $(Z_k)_i$  est inhérent à l’aspect incrémentiel de la construction du vocabulaire :

$$(Z_k)_0 \subseteq (Z_k)_1 \subseteq \dots \subseteq (Z_k)_{i-1} \subseteq (Z_k)_i$$

avec  $(Z_k)_0 = \emptyset$  (les primitives de l’espace de représentation  $k$  extraites dans l’image  $I_i$  sont utilisées pour obtenir  $(Z_k)_{i+1}$ ). De plus, soit  $(z_k)_i$  le sous-ensemble des mots de  $(Z_k)_i$  qui caractérisent l’image  $I_i$  dans l’espace de représentation  $k$  (i.e., ce sont les mots de  $(Z_k)_i$  qui sont trouvés dans  $I_i$ ) :

$$I_i \Leftrightarrow (z_k)_i \quad \text{avec } (z_k)_i \subseteq (Z_k)_i$$

Étant donné que plusieurs espaces de représentation sont pris en compte ici, différentes caractérisations d’une même image sont possibles (i.e., une par espace de représentation). Ainsi, soit  $(z^n)_i = (z_0)_i, \dots, (z_n)_i$  la représentation globale de l’image  $I_i$ , tous espaces de représentation  $k = 0, \dots, n$  confondus. La probabi-



lité a posteriori, qui peut maintenant s'écrire  $p(S_t|(z^n)_t, M_{t-1})$ , peut alors être exprimée comme suit :

$$p(S_t|(z^n)_t, M_{t-1}) = \eta p((z^n)_t|S_t, M_{t-1}) p(S_t|M_{t-1}) \quad (2.3)$$

En tenant compte des corrélations existant entre les différents dictionnaires, il serait possible d'obtenir des informations additionnelles sur l'occurrence des mots. Cependant, en faisant l'hypothèse que les espaces de représentation des images sont indépendants, on peut obtenir une formulation mathématique pratique pour l'estimation de la probabilité a posteriori :

$$p(S_t|(z^n)_t, M_{t-1}) = \eta \left[ \prod_{k=0}^n p((z_k)_t|S_t, M_{t-1}) \right] p(S_t|M_{t-1}) \quad (2.4)$$

où la probabilité conditionnelle  $p((z_k)_t|S_t, M_{t-1})$  est considérée comme une fonction de vraisemblance  $\mathcal{L}(S_t|(z_k)_t, M_{t-1})$  de  $S_t$  : étant donné le modèle de l'environnement  $M_{t-1}$ , on évalue, pour chacun des événements  $S_t = i$ , la vraisemblance des mots  $(z_k)_t$  (voir section 2.3.2).

Il est finalement possible d'estimer la probabilité a posteriori de façon récursive en décomposant la partie droite de l'équation 2.4 comme suit :

$$p(S_t|(z^n)_t, M_{t-1}) = \eta \left[ \prod_{k=0}^n p((z_k)_t|S_t, M_{t-1}) \right] \underbrace{\sum_{i=-1}^m p(S_t|S_{t-1} = i, M_{t-1}) p(S_{t-1} = i|M_{t-1})}_{\text{prédiction}} \quad (2.5)$$

où  $p(S_t|S_{t-1} = i, M_{t-1})$  est le modèle d'évolution temporelle de notre filtre Bayésien discret (voir section 2.3.1). Le caractère récursif de la formulation mathématique proposée dans l'équation 2.5 vient du fait que  $p(S_{t-1}|M_{t-1})$  est une écriture factorisée de  $p(S_{t-1}|(z^n)_{t-1}, M_{t-2})$ , la probabilité a posteriori calculée à l'instant  $t - 1$ , compte tenu de la mise à jour du modèle de l'environnement  $M_{t-2}$  avec  $(z^n)_{t-1}$  pour former  $M_{t-1}$ .

D'après l'équation 2.5, on s'aperçoit que l'estimation de la probabilité a posteriori à l'instant  $t$  requiert dans un premier temps l'intégration temporelle de cette même quantité obtenue au temps  $t - 1$ , et considérée ici comme la *probabilité a priori*, afin d'obtenir ce que nous appellerons dans la suite de ce mémoire la *prédiction* à l'instant  $t$ . Cette prédiction est ensuite multipliée successivement par les probabilités conditionnelles correspondant aux vraisemblances obtenues dans les différents espaces de représentation de l'image courante, afin de confirmer ou d'infirmer la prédiction.

Il est important de noter ici que dans le cadre de notre filtre Bayésien discret, l'ensemble des lieux formant le modèle  $M_t$  de l'environnement évolue dans le temps avec l'acquisition de nouvelles images, divergeant du cadre classique du filtrage Bayésien où cet ensemble serait statique.

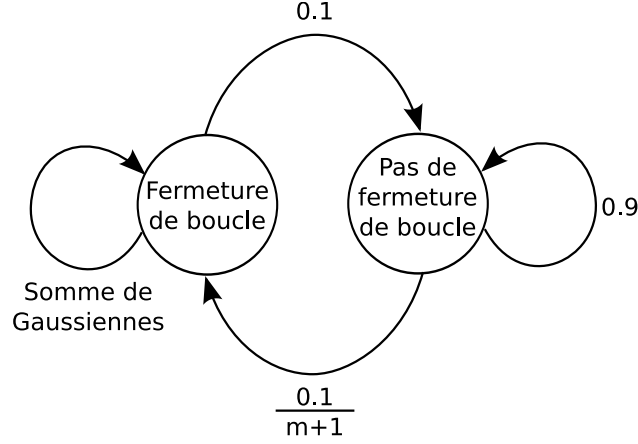


FIG. 2.9: Modèle d'évolution temporelle représenté sous la forme d'un graphe d'états. Le modèle proposé ici peut être qualifié de "stationnaire" : la probabilité de rester dans le même état est plus forte que la probabilité de changer d'état.

### 2.3.1 Modèle d'évolution temporelle

La formulation récursive de la probabilité a posteriori proposée dans l'équation 2.5 permet d'obtenir la prédiction à l'instant  $t$  à partir de la probabilité a priori au même instant. Pour cela, la probabilité a priori est intégrée dans le temps sur toutes les transitions possibles entre  $t - 1$  et  $t$  grâce au modèle d'évolution temporelle du filtre Bayésien discret : ce modèle donne la probabilité  $p(S_t = i_2 | S_{t-1} = i_1, M_{t-1})$  d'être dans l'état  $i_2$  à l'instant  $t$  considérant que l'état précédent était  $i_1$ .

L'enjeu ici est de pouvoir modéliser fidèlement les transitions possibles entre lieux et susceptibles d'être effectuées par la caméra entre deux images consécutives : il s'agit de prédire le lieu de la prochaine image. Le modèle d'évolution temporelle joue donc un rôle crucial dans le processus d'estimation, car il permet d'en assurer la cohérence temporelle. Il s'avère de ce fait déterminant pour filtrer les erreurs dues à des ressemblances éphémères entre l'image courante et un lieu du modèle de l'environnement : si cette ressemblance n'est pas maintenue au cours du temps dans plusieurs images consécutives, l'hypothèse correspondante aura peu de chances d'être soutenue et d'avoir une probabilité élevée.

En fonction des valeurs respectives de  $S_{t-1}$  et  $S_t$ , la probabilité de transition est définie comme suit (voir également figure 2.9) :

- $p(S_t = -1 | S_{t-1} = -1, M_{t-1}) = 0.9$ , la probabilité qu'aucune fermeture de boucle ne se produise à l'instant  $t$  est importante compte tenu du fait qu'aucune ne s'est produite au pas de temps précédent (i.e., la probabilité de rester dans l'évènement "pas de fermeture de boucle" est élevée)
- $p(S_t = i_2 | S_{t-1} = -1, M_{t-1}) = \frac{0.1}{m+1}$  avec  $i_2 \in [0; m]$ , la probabilité d'une fermeture de boucle à l'instant  $t$  est faible compte tenu qu'aucune ne s'est produite à l'instant  $t - 1$
- $p(S_t = -1 | S_{t-1} = i_1, M_{t-1}) = 0.1$  avec  $i_1 \in [0; m]$ , la probabilité de l'évènement "pas de

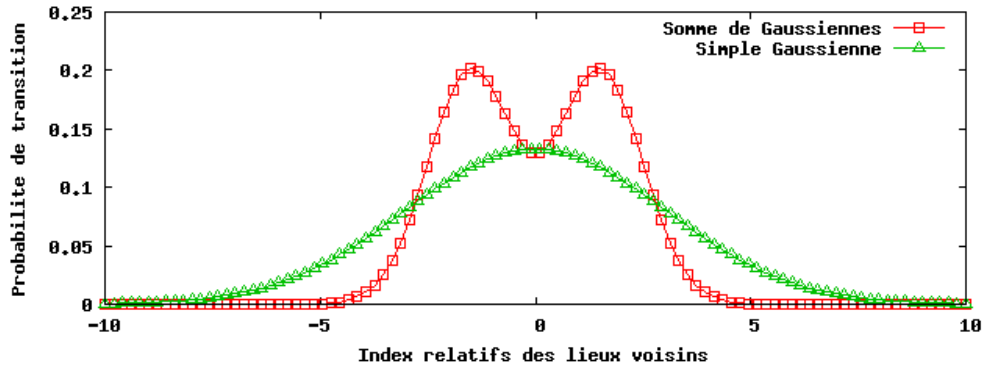


FIG. 2.10: Somme de Gaussiennes pour le modèle d'évolution. D'après la stratégie de sélection d'images décrite dans la section 2.2, une image n'est retenue pour le calcul de la probabilité de fermeture de boucle que si elle diffère suffisamment de la dernière image traitée pour considérer que la caméra s'est déplacée. En conséquence, une importance accrue est nécessaire sur les états voisins de l'état considéré plutôt que sur celui-ci en particulier. La probabilité de transition correspondante est de fait mieux modélisée par une somme de Gaussiennes (trait plein) que par une seule Gaussienne (tirets). Cependant, sans le mécanisme de sélection automatique des images, le modèle de Gaussienne simple serait plus approprié.

fermeture de boucle à l'instant  $t$  est faible étant donné qu'une fermeture de boucle s'est produite au pas de temps précédent

- $p(S_t = i_2 | S_{t-1} = i_1, M_{t-1})$ , avec  $i_1, i_2 \in [0; m]$ , est une somme de deux Gaussiennes centrées sur les états  $i_1 = i_2 - 1$  and  $i_1 = i_2 + 1$  quand on considère l'état  $i_2$  : la stratégie de sélection des images (voir section 2.2) nous permet ici d'assumer ici que la probabilité de rester dans l'état  $i_1 = i_2$  est plus faible que la probabilité de se retrouver dans un état voisin (voir figure 2.10). Dans le modèle de somme de Gaussiennes proposé ci-dessus, seuls les états  $i_1 = i_2 - 3, \dots, i_2 + 3$  ont une probabilité non-négligeable. La taille de ce voisinage peut être adapté en fonction de la vitesse de déplacement de la caméra et de la fréquence d'acquisition des images.

Il est important de noter ici que pour que la probabilité  $p(S_t \geq -1 | S_{t-1} = i_1, M_{t-1})$  vaille 1 lorsque  $i_1 \in [0; m]$ , il faut que les coefficients de la somme de Gaussiennes employée dans le dernier cas somment à 0.9.

### 2.3.2 Système de vote pour l'estimation de la vraisemblance

Ainsi que cela a déjà été remarqué précédemment lors de la formulation mathématique de la détection de fermeture de boucle (voir équation 2.5), l'estimation de la probabilité a posteriori nécessite le calcul de la fonction de vraisemblance  $\mathcal{L}(S_t | (z_k)_t, M_{t-1})$  pour chaque espace de représentation  $k$ . Étant donné que les espaces de représentation sont considérés indépendants, il est possible d'appliquer le modèle de vraisemblance décrit ci-après à chacun d'entre eux séparément.

Dans le cadre du filtrage Bayésien, la vraisemblance peut être définie comme une mesure de l'adéquation des observations avec le modèle estimé : pour chaque entrée de ce dernier, on prédit l'observation la plus vraisemblable compte tenu des caractéristiques de cette entrée, puis on la compare à l'observation réelle. Lorsque cette comparaison s'avère positive, l'hypothèse correspondante apparaît alors comme plausible. Dans une approche Bayésienne naïve, le modèle de vraisemblance consisterait ici à évaluer la probabilité d'observer chaque mot du dictionnaire dans chacun des lieux du modèle (i.e., par le comptage du nombre d'occurrences correspondantes). Cela reviendrait à considérer les probabilités d'occurrence des mots comme totalement indépendantes. Pour un lieu  $L_i$  en particulier, ceci donnerait donc :

$$p\left((z_k)_t | S_t = i, M_{t-1}\right) = \prod_{w \in (z_k)_t} p(w | S_t = i, M_{t-1}), \quad \text{avec } p(w | S_t = i, M_{t-1}) = \frac{n_{wi}}{n_i} \quad (2.6)$$

où  $n_{wi}$  désigne le nombre d'occurrences du mot  $w$  dans le lieu  $L_i$ , alors que  $n_i$  est le nombre total d'occurrences de mots dans  $L_i$ . Ainsi, la vraisemblance d'un lieu à l'instant  $t$  est conditionnée par l'occurrence dans ce lieu des mots actuellement observés : plus ces mots apparaissent fréquemment dans le lieu considéré, plus la vraisemblance correspondante sera élevée. On peut remarquer que l'approche Bayésienne naïve ne prend pas en compte la répartition des mots dans les différents lieux du modèle de l'environnement : peu importe le nombre de lieux dans lesquels un mot a été perçu, sa contribution à l'estimation de la vraisemblance d'un lieu ne dépend que de son occurrence dans ce lieu. Pourtant, il peut être judicieux de prendre en compte cette information de répartition, étant donné qu'un mot vu dans tous les lieux de l'environnement n'est pas discriminant.

Dans ce mémoire, nous proposons comme alternative à l'approche Bayésienne naïve pour l'estimation de la vraisemblance une méthode heuristique inspirée des travaux de [Filliat, 2007] : l'auteur définit un système de vote simple pour la tâche de localisation globale visuelle. Nous avons adapté ce système de vote à la formulation Bayésienne mise en oeuvre ici pour en faire notre modèle de vraisemblance, en veillant à considérer le caractère discriminant des mots et leur répartition dans les différents lieux de l'environnement. D'une manière générale, les méthodes de vote évaluent les différentes hypothèses sur la base d'une note. Dans [Filliat, 2007], cette note est incrémentée sur la base de la pertinence des mots perçus dans l'image courante compte tenu de l'hypothèse considérée.

### La méthode de vote

La fonction de vraisemblance  $\mathcal{L}\left(S_t | (z_k)_t, M_{t-1}\right)$  permet de déterminer, parmi tous lieux  $L_i$  du modèle  $M_{t-1}$ , ceux dont la caractérisation est similaire à l'image courante dans l'espace de représentation  $k$ . Afin d'éviter une comparaison exhaustive et computationnellement chère des mots de l'image courante avec ceux de l'ensemble des lieux du modèle, nous utilisons un index inversé qui contient, pour chaque mot du

dictionnaire  $k$ , les lieux dans lesquels celui-ci a été perçu. Ainsi, grâce à cet index inversé, il est possible d'implémenter une simple procédure de vote pour estimer la similarité des lieux du modèle avec l'image courante : lorsqu'un mot visuel est extrait dans cette image, l'index inversé permet d'incrémenter une note originellement nulle pour l'ensemble des lieux dans lesquels ce mot a été vu (voir figure 2.11). La quantité utilisée pour l'incrément est inspirée du coefficient *tf-idf* (*term frequency – inverted document frequency*, [Sivic and Zisserman, 2003]) :

$$\text{tf-idf} = \frac{n_{wi}}{n_i} \log \frac{N}{n_w} \quad (2.7)$$

où  $n_{wi}$  est le nombre d'occurrences du mot  $w$  dans le lieu  $L_i$ ,  $n_i$  est le nombre total d'occurrences de mots recensées dans le lieu  $L_i$ ,  $n_w$  est le nombre de lieux contenant le mot  $w$ , et  $N$  est le nombre total de lieux dans le modèle. D'après l'équation 2.7, on constate que le coefficient *tf-idf* est le résultat du produit de la fréquence du mot dans un lieu donné par l'inverse de la fréquence des lieux contenant ce mot. Ce coefficient est calculé chaque fois que la fonction de vraisemblance est évaluée, à partir des statistiques les plus récentes sur les mots du dictionnaire et les lieux du modèle de l'environnement. Il permet notamment de donner de l'importance aux mots vus un grand nombre de fois dans un nombre limité de lieux, et de pénaliser à l'inverse les mots trop courants qui sont vus partout : le coefficient *tf-idf* permet de prendre en compte le pouvoir discriminant des mots. Ceci est particulièrement remarquable dans l'étude des caractéristiques du coefficient *tf-idf* aux conditions limites :

- Cas 1 : mot  $w$  vu une seule fois en tout, dans un seul lieu  $L_w$  parmi 50

$$\text{tf-idf}(L_i) = \begin{cases} \frac{1}{n_i} \log \frac{50}{1} & \text{si } L_i = L_w \\ 0 & \text{autrement (i.e., } n_{wi} = 0 \quad \forall L_i \neq L_w) \end{cases}$$

- Cas 2 : mot  $w$  vu 200 fois en tout, dans un seul lieu  $L_w$  parmi 50

$$\text{tf-idf}(L_i) = \begin{cases} \frac{200}{n_i} \log \frac{50}{1} & \text{si } L_i = L_w \\ 0 & \text{autrement (i.e., } n_{wi} = 0 \quad \forall L_i \neq L_w) \end{cases}$$

- Cas 3 : mot  $w$  vu une fois dans chacun des 50 lieux

$$\text{tf-idf}(L_i) = \frac{1}{n_i} \log \frac{50}{50} = 0 \quad \forall L_i$$

- Cas 4 : mot  $w$  vu 200 fois dans chacun des 50 lieux

$$\text{tf-idf}(L_i) = \frac{200}{n_i} \log \frac{50}{50} = 0 \quad \forall L_i$$

D'après l'analyse des caractéristiques du coefficient *tf-idf* aux conditions limites, on observe que dans

les situations où un mot n’apporte aucune information (i.e., lorsqu’il est vu partout, cas 3 et 4), sa contribution au vote est nulle. En revanche, lorsqu’un mot n’est vu que dans un lieu en particulier, c’est son occurrence dans ce lieu qui permet de pondérer sa contribution. Ainsi, dans le cas 1 la contribution sera moins importante que dans le cas 2 si on considère le même lieu :

$$\frac{1}{n_i} \log \frac{50}{1} < \frac{200}{n_i} \log \frac{50}{1}$$

Avant de détailler plus en avant le reste de notre modèle de probabilité, nous allons faire un parallèle entre le mécanisme de vote présenté ci-dessus et l’approche Bayésienne naïve pour l’estimation de la vraisemblance. Partant de l’équation 2.6, on peut écrire la *log-vraisemblance* pour un lieu  $L_i$  donné comme suit :

$$\log \left( p \left( (z_k)_t | S_t = i, M_{t-1} \right) \right) = \sum_{w \in (z_k)_t} \log \left( p(w | S_t = i, M_{t-1}) \right) = \sum_{w \in (z_k)_t} \log \left( \frac{n_{wi}}{n_i} \right) \quad (2.8)$$

On s’aperçoit alors que la vraisemblance pour un lieu en particulier peut être calculée sur la base d’une somme dont les éléments sont des statistiques sur les mots du dictionnaire qui sont présents dans l’image courante. La méthode de vote que nous proposons réalise le même genre de somme, mais en prenant en compte des statistiques plus pertinentes que les simples occurrences de mots :

$$p \left( (z_k)_t | S_t = i, M_{t-1} \right) = \sum_{w \in (z_k)_t} \frac{n_{wi}}{n_i} \log \frac{N}{n_w} \quad (2.9)$$

En résumé, lorsqu’un mot est vu dans l’image courante, les lieux où il a déjà été perçu par le passé voient leur note mise à jour grâce au coefficient tf-idf associé à la paire  $\{mot-lieu\}$ . La mise à jour consiste en l’ajout de statistiques à propos de l’occurrence de ce mot dans le lieu considéré.

### Évènement “pas de fermeture de boucle à l’instant $t$ ”

La méthode simple de vote introduite ci-dessus permet d’obtenir, pour chaque lieu du modèle, une note indiquant son niveau de similarité avec l’image courante. Cependant, il faut également évaluer la vraisemblance de l’évènement “pas de fermeture de boucle à l’instant  $t$ ”. Cet évènement permet de gérer les situations où l’image courante ne ressemble particulièrement à aucun des lieux du modèle et vient de ce fait probablement d’un nouveau lieu. Afin d’évaluer la vraisemblance de cet évènement, nous introduisons dans le modèle un lieu virtuel,  $L_{-1}$ , dans le but de lui accorder une note élevée en l’absence de fermeture de boucle. Lors de chaque évaluation de la fonction de vraisemblance, le lieu virtuel est construit à partir des mots les plus fréquemment vus du dictionnaire et ajouté au modèle de l’environnement. Cette simple construction permet d’obtenir le comportement souhaité en fonction de la provenance de l’image courante.

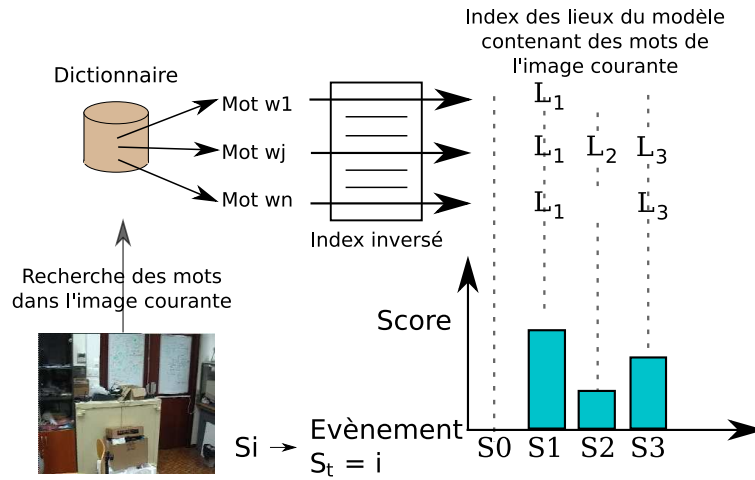


FIG. 2.11: Système de vote : la liste des lieux dans lesquels les mots visuels de l'image courante ont été vus est obtenue à partir de l'index inversé puis utilisée pour l'estimation de la fonction de vraisemblance.

En cas d'absence de fermeture de boucle,  $I_t$  sera statistiquement plus semblable à  $L_{-1}$  que n'importe quel autre lieu du modèle. A l'inverse, si l'image courante vient d'un lieu  $L_i$  existant, les mots visuels responsables de la fermeture de boucle correspondante seront trouvés uniquement en ce lieu et dans  $I_t$ , faisant d'eux des mots peu fréquents et improbables dans la composition de  $L_{-1}$ . Par ailleurs, en cas d'aliasing perceptuel (i.e., lorsque l'image courante ressemble à plusieurs lieux distincts), différents lieux recevront une note élevée. Afin d'éviter une détection erronée de fermeture de boucle, il est important que dans ce cas  $L_{-1}$  reçoive un score du même ordre de grandeur que ces lieux. Avec le mécanisme proposé ici,  $L_{-1}$  sera en partie composé des mots responsables de l'aliasing perceptuel, étant donné que ces mots se retrouvent dans plusieurs lieux différents (i.e., ce sont donc des mots fréquents). En conséquence,  $L_{-1}$  sera doté d'un score raisonnable, sans pour autant bénéficier de la note la plus élevée.

La construction d'un lieu virtuel à partir de mots existant est comparable à la génération de lieux par tirages aléatoires de mots proposée par les auteurs de [Cummins and Newman, 2007]. L'existence d'un lieu virtuel peut être simulé simplement ici en ajoutant une entrée  $L_{-1}$  à l'index inversé pour chacun des mots les plus fréquemment vus. Ainsi, si l'un d'entre eux est trouvé dans  $I_t$ , il votera pour  $L_{-1}$  comme décrit dans la figure 2.12, exactement de la même manière que pour les autres lieux.

### Mise à jour de la prédiction

Une fois que tous les mots ont voté pour les lieux auxquels ils appartiennent, il faut mettre à jour la distribution de probabilité que représente la prédiction. Plutôt que de multiplier la probabilité de chaque lieu du modèle par sa note après le vote, on ne retient ici que les lieux les plus vraisemblables pour la mise à jour, ce qui revient à considérer que les autres lieux ont leur probabilité multipliée par 1. Ainsi, seules les

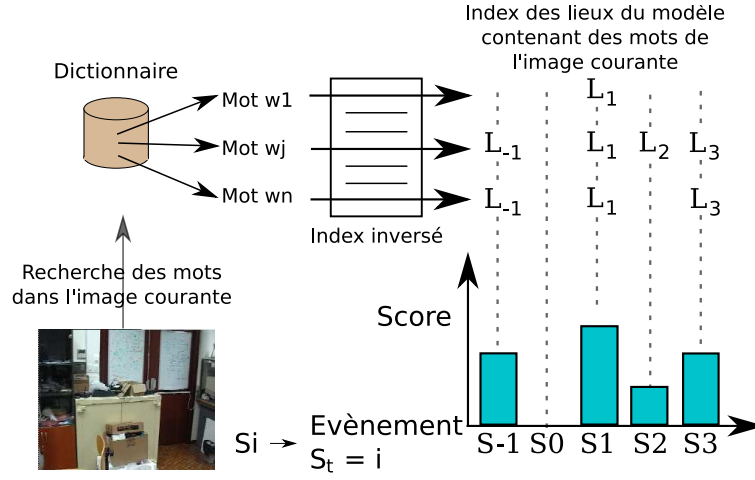


FIG. 2.12: Prise en compte du lieu virtuel : celui-ci est composé des mots les plus fréquemment vus du dictionnaires et la vraisemblance correspondante peut être simplement calculée comme pour les lieux "réels".

hypotheses dont le score est significativement plus élevé que la moyenne des scores sont considérées ici pour la mise à jour. Exprimé mathématiquement, cela revient à sélectionner les hypotheses dont le *coefficient de variation particulier* (i.e., le c.o.v. particulier, l'écart à la moyenne de la note normalisé par la moyenne) est supérieur au c.o.v. standard (i.e., l'écart type normalisé par la moyenne). D'autre part, la quantité utilisée pour la mise à jour de la probabilité a posteriori associée à un lieu  $L_i$  est dans ce cas donné par la différence entre le c.o.v. particulier correspondant et le c.o.v. standard, plus 1 (ce qui, après simplification, correspond à la différence entre le score  $s_i$  du lieu et l'écart type  $\sigma$ , normalisé par la moyenne  $\mu$ ) :

$$\mathcal{L}(S_t = i | (z_k)_t, M_{t-1}) = \begin{cases} \frac{s_i - \mu}{\mu} - \frac{\sigma}{\mu} + 1 = \frac{s_i - \sigma}{\mu} & \text{si } s_i \geq \mu + \sigma \\ 1 & \text{autrement} \end{cases} \quad (2.10)$$

La quantité obtenue par l'équation 2.10 correspond donc à la valeur de la fonction de vraisemblance  $\mathcal{L}(S_t = i | (z_k)_t, M_{t-1})$  pour l'hypothèse  $i$  et sert à la mise à jour de la prédiction (voir 2.13). Il est important que cette quantité soit normalisée, de manière à empêcher qu'un espace de représentation réalise des mises à jour d'un ordre de grandeur différent des autres. Lorsque tous les lieux le nécessitant ont été mis à jour, la prédiction est normalisée afin de représenter la nouvelle distribution de probabilité a posteriori de fermeture de boucle.

### 2.3.3 Gestion des hypothèses a posteriori

Une fois la probabilité a posteriori mise à jour avec les modèles de vraisemblance et normalisée, il faut sélectionner, s'il existe, le lieu du modèle dont la probabilité est assez élevée pour considérer qu'il



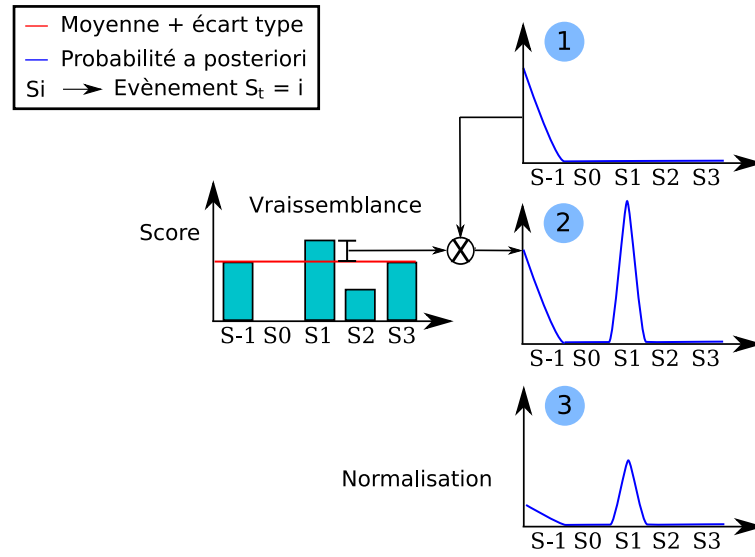


FIG. 2.13: Mise à jour de la prédiction par la vraisemblance : lorsque la vraisemblance d'un lieu est supérieure à la somme de la moyenne et de l'écart type, la probabilité correspondante dans la prédiction (cadre "1") est mise à jour (cadre "2"). Une fois toutes les mises à jour effectuées, la probabilité est normalisée (cadre "3").

correspond à une hypothèse plausible de fermeture de boucle. Le seuil choisi ici pour détecter de tels candidats est fixé à 0.8. Cependant, il est peu probable que la distribution de probabilité a posteriori exhibe un pic prononcé pour une hypothèse en particulier. Cette distribution sera plutôt en effet répartie sur un ensemble restreint de lieux voisins en raison des similarités qu'ils sont susceptibles de partager : il n'y a pas de délimitation brutale entre les lieux, et lorsqu'ils sont voisins, ceux-ci possèdent un certain nombre de caractéristiques communes. Cela est principalement dû à la nature consécutive de l'acquisition des images et à leurs similarités lorsqu'elles sont proches dans le temps. Il s'agit par ailleurs d'une caractéristique fondamentale du modèle de filtrage Bayésien employé ici : ainsi que nous l'avons déjà exposé précédemment (voir section 2.3.1), il est important d'assurer la cohérence temporelle de l'estimation. Ainsi, on sélectionne comme hypothèse probable de fermeture de boucle tout lieu pour lequel la somme des probabilités sur les lieux voisins est supérieure au seuil fixé. La taille du voisinage pris en compte ici est la même que celle proposée dans la somme de Gaussiennes utilisée dans le modèle d'évolution temporelle (voir section 2.3.1).

Lorsqu'une hypothèse plausible est trouvée, au sens du critère de sélection décrit ci-dessus, un algorithme de géométrie multi-vues [Nistér, 2004] est employé afin de vérifier que la contrainte de géométrie épipolaire [Hartley and Zisserman, 2004] est satisfaite. Cela permet d'assurer l'existence d'une structure commune et d'un changement de point de vue cohérent (i.e., une transformation composée d'une rotation et d'une translation) entre l'image courante et le lieu présumé de fermeture de boucle avant de valider définitivement l'hypothèse : il peut arriver en effet qu'en cas d'aliasing perceptuel,  $I_t$  ressemble fortement à un lieu du modèle sans pour autant en provenir, auquel cas la validation de la contrainte de géométrie

épipolaire permet de lever l'ambiguïté. Pour ce faire, la méthode employée ici repose sur une procédure RANSAC [Fischler and Bolles, 1981] afin de générer un ensemble de transformations plausibles entre  $I_t$  et le lieu considéré sur la base d'appariements de primitives SIFT, selon l'algorithme des cinq points [Nistér, 2004]. Les transformations incohérentes sont alors rejetées par l'imposition d'un seuil sur l'erreur moyenne de re-projection : à partir d'une transformation donnée, les primitives SIFT sont projetées de l'image courante vers une vue du lieu considéré, la moyenne des distances Euclidiennes séparant les positions projetées des positions 2D initiales dans cette vue correspondant à l'erreur de moyenne re-projection. Le processus de détermination de la transformation entre les points de vue de deux images est illustré dans la figure 2.14. Il est important de noter que les primitives SIFT utilisées dans cette procédure ont déjà été extraites pour la caractérisation sous la forme de sacs de mots visuels, il n'est donc pas nécessaire de répéter cette étape ici.

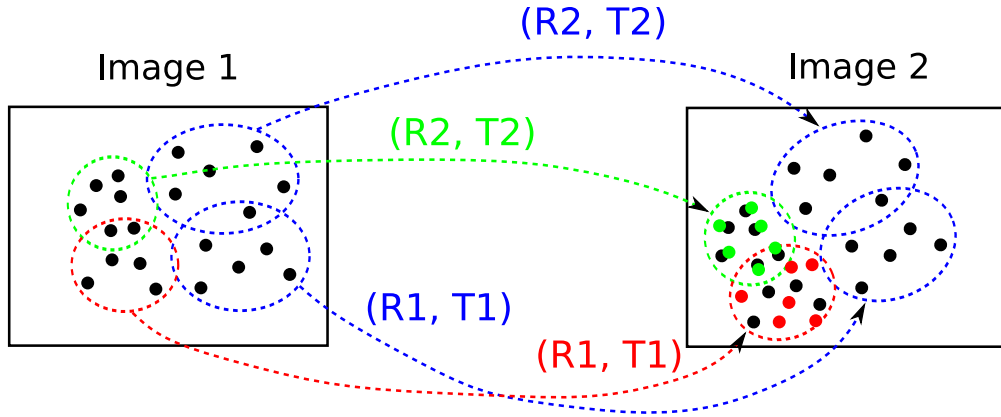


FIG. 2.14: Illustration de la procédure permettant de déterminer (si elle existe) la transformation joignant les points de vue de deux images. A partir de l'appariement d'un sous-ensemble de primitives locales entre les deux images (ellipse bleue), une première transformation  $(R1, T1)$  est calculée. Celle-ci est utilisée pour essayer de projeter d'autres primitives (ellipse rouge) d'une image à l'autre : les projections (points rouges) sont trop éloignées de leurs correspondants dans la seconde image, la transformation est rejetée. Une seconde transformation  $(R2, T2)$  est calculée à partir d'un nouveau sous-ensemble de primitives, permettant cette fois-ci des projections cohérentes (points verts) dans l'autre image : la transformation est retenue.

Une hypothèse plausible pour laquelle la contrainte de géométrie épipolaire est validée est finalement acceptée et le lieu correspondant est considéré comme lieu de fermeture de boucle. Dans ce cas, la caractérisation de ce lieu est augmentée par les mots visuels provenant de l'image courante. Dans le cas contraire, un nouveau lieu décrit par ces mêmes mots est ajouté au modèle. Cependant, si une hypothèse plausible a été rejetée par la géométrie multi-vues, sa probabilité ne va pas pour autant être nulle à l'estimation suivante : celle-ci va progressivement être diffusée sur les lieux voisins et décroître sur plusieurs pas de temps, en fonction des critères de la somme de Gaussiennes utilisée dans le modèle d'évolution temporelle (voir section 2.3.1). Ainsi, les hypothèses correctes qui sont rejetées de façon erronée par l'algorithme de géométrie multi-vues seront toujours supportées dans les instants futurs proches, jusqu'à ce qu'une transformation

cohérente soit trouvée.

## 2.4 Mise en cache d'hypothèses

Une fois qu'une image a été complètement traitée (i.e., si elle n'a pas été écartée en raison d'une similarité trop importante avec la dernière image considérée, voir figure 2.1), soit un nouveau lieu doit être ajouté au modèle, soit un lieu existant doit être mis à jour, en fonction de la distribution de probabilité a posteriori (voir section 2.3). Dans les deux cas, l'information visuelle provenant de l'image courante est provisoirement gardée en cache avant de servir à la mise à jour du modèle. Pendant ce temps, cette information n'est pas prise en compte pour les détections de fermeture de boucle ultérieures, étant donné qu'elle n'a pas été effectivement ajoutée au modèle. Ce mécanisme de cache permet d'éviter des détections locales de fermeture de boucle dues aux similarités qui existent entre les images voisines dans le temps, comme expliqué ci-après.

La stratégie de sélection des images permet d'écarter les images dont le taux de similarité est proche de 100% (voir section 2.2). Cependant, pour assurer la cohérence temporelle de l'estimation, les images acquises consécutivement et prises en compte pour le calcul de la probabilité de fermeture de boucle doivent posséder un certain nombre de caractéristiques communes, comme cela a été souligné dans la section 2.3.1. Ainsi, après la mise à jour du modèle de l'environnement par l'ajout ou l'augmentation d'un lieu, la prochaine image acquise partagera des points communs avec ce lieu, générant en conséquence un score élevé pour ce dernier lors du calcul de la vraisemblance : la probabilité d'une fermeture de boucle avec ce lieu s'en trouvera donc favorisée. Toutefois, l'importante note accordée à l'hypothèse correspondante tient uniquement aux similarités dues à la proximité dans le temps entre l'image et le lieu considérés. Encore une fois, il est indispensable que ces similarités existent entre les images acquises et, par conséquent, entre les lieux correspondants. Généraliser la stratégie de sélection des images pour écarter toute image possédant ne serait-ce qu'un faible taux de similarité avec le dernier lieu mis à jour dans le modèle n'est donc pas envisageable : cela reviendrait finalement à segmenter les lieux de manière brutale, interdisant de fait tous points communs entre lieux voisins, allant à l'encontre du principe de continuité temporelle dans l'apparence des images traitées.

En conséquence, choisir d'écarter les images même faiblement semblables afin d'éviter les détections locales de fermeture de boucle aurait un impact négatif sur la qualité de l'estimation. C'est pourquoi nous avons mis en place ici un système de cache garantissant la continuité temporelle de l'apparence et interdisant les détections locales de fermeture de boucle. Ainsi, chaque hypothèse est gardée en cache tant que sa mesure de similarité (voir section 2.2) avec l'image actuellement considérée est au dessus d'un certain seuil (i.e., 20% dans les expériences rapportées dans la suite de ce mémoire). Le mécanisme de cache est illustré dans la figure 2.15.

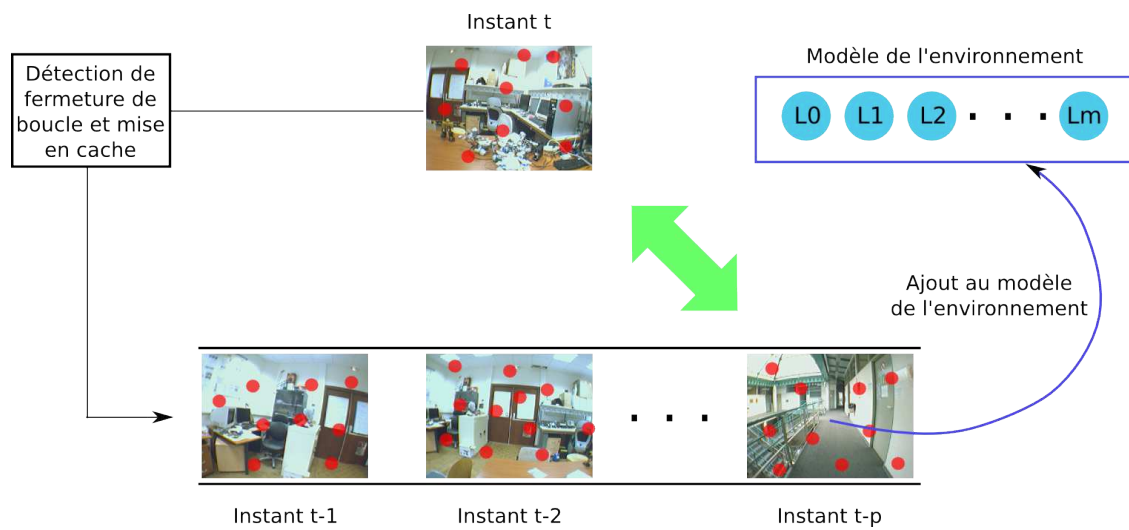


FIG. 2.15: Illustration du mécanisme de cache. Après avoir servi à la détection de fermeture de boucle, l'information visuelle de l'image courante (i.e., instant  $t$ ) est mise en cache. Avant cela, cette information est comparée au plus ancien élément (i.e., instant  $t - p$ ) de la file constituant ce cache. Si la mesure de similarité résultante est en-deçà de 20%, cet élément est retiré du cache pour être ajouté au modèle de l'environnement, servant soit à créer un nouveau lieu, soit à augmenter la description d'un des lieux déjà présents.

## Chapitre 3

# Résultats expérimentaux

Dans ce chapitre nous présentons une série de résultats expérimentaux destinés à valider l’approche développée dans le cadre des travaux rapportés dans ce mémoire. Pour cela, différentes séquences vidéos ont été acquises dans des environnements d’intérieur et d’extérieur avec une simple caméra monoculaire tenue à la main. Lors de l’acquisition des images, plusieurs cycles ont été réalisés pour la trajectoire de la caméra. Ainsi, il est possible de vérifier que l’algorithme mis en oeuvre détecte bien les fermetures de boucle correspondantes. De plus, en estimant “manuellement” la trajectoire de la caméra, sur la base des images acquises, il est possible d’évaluer la vérité terrain permettant de déterminer les images correspondant à des fermetures de boucle. Cette vérité terrain offre alors la possibilité d’évaluer quantitativement les performances de reconnaissance de lieu de notre algorithme.

Comme indiqué dans la section 2.1.2, plusieurs espaces de représentation peuvent être utilisés dans la solution que nous proposons. Afin de mieux percevoir l’utilité de la combinaison des caractérisations d’image, nous donnons dans les résultats expérimentaux présentés ci-après une comparaison des résultats obtenus avec et sans la prise en compte de ces différentes sources d’information. C’est pourquoi ce chapitre est découpé en trois sections. Les deux premières d’entre elles donnent les résultats expérimentaux obtenus sur des séquences d’images provenant d’environnements d’intérieur (première section) et d’extérieur (deuxième section). Dans chacune de ces sections, les expériences ont été réalisées en combinant les primitives SIFT aux histogrammes locaux de teinte. Ensuite, dans la dernière section, nous présentons une évaluation de l’influence de ces espaces de représentation sur la qualité de l’estimation et sur les temps de calcul.

Les résultats expérimentaux rapportés ici ont fait l’objet de plusieurs publications ([Angeli et al., 2008a], [Angeli et al., 2008b]) et des vidéos en démontrant la qualité sont disponibles à l’adresse suivante : <http://animatlab.lip6.fr/AngeliVideosFr>



FIG. 3.1: Exemples d'images provenant de la séquence d'intérieur : les images sont ordonnées par ordre d'acquisition, suivant le déplacement de la caméra lors de l'expérience.

### 3.1 Environnement d'intérieur

La première section de ce chapitre est dédiée aux résultats obtenus à partir d'une séquence vidéo réalisée dans un environnement d'intérieur (cf. figure 3.1 pour un aperçu des images provenant de la séquence). Cette séquence a été réalisée à partir d'un simple caméscope tenu à la main, celui-ci présentant un angle de vue de  $60^\circ$  et une gestion automatique de l'exposition. Pour cette expérience, l'acquisition des images a été effectuée à 1Hz, avec une taille de  $240 \times 192$  pixels.

La trajectoire suivie par la caméra au cours de cette expérience est visible dans la figure 3.2. Dans cette figure, trois couleurs différentes sont utilisées pour représenter cette trajectoire. Lorsque la probabilité a posteriori est en deçà du seuil fixé pour la détection de fermeture de boucle (voir section 2.3.3), la couleur retenue est le bleu. Si cette probabilité dépasse le seuil fixé, et que la contrainte de géométrie épipolaire est satisfaite, la trajectoire devient verte : une fermeture de boucle a été détectée. Enfin, si la probabilité est au dessus du seuil, mais que la contrainte de géométrie épipolaire n'est pas satisfaite, cela signifie qu'une hypothèse probable de fermeture de boucle a été rejetée : la trajectoire apparaît alors en rouge.

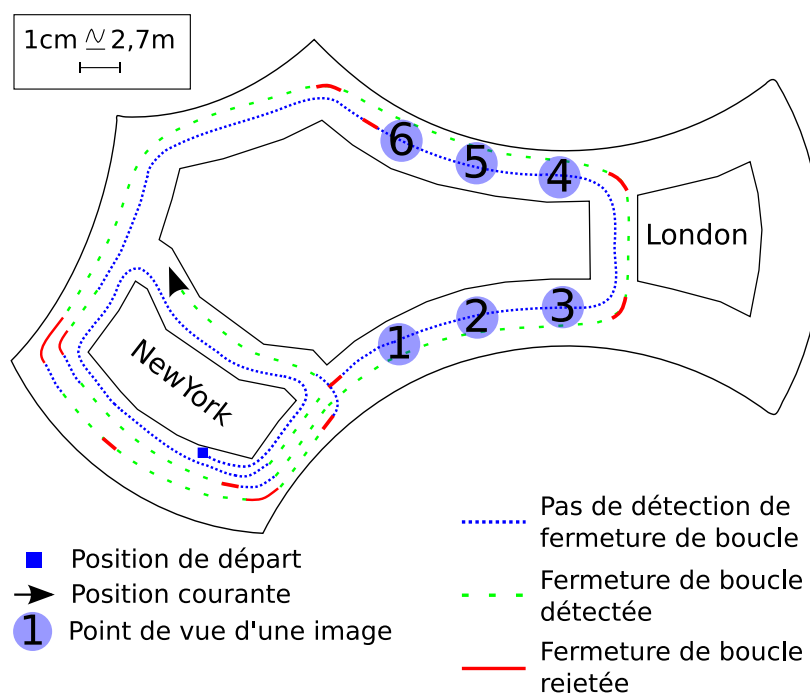


FIG. 3.2: Trajectoire complète de la caméra à la fin de l'expérience en environnement d'intérieur. Une première petite boucle est réalisée autour des ascenseurs "New York" sur la gauche avant de rejoindre les ascenseurs "London" sur la droite, empruntant pour ce faire le couloir du bas dans le plan. Une partie de la petite boucle est alors effectuée une seconde fois, lorsque la caméra revient des ascenseurs "London" par le couloir du haut dans le plan. Ensuite, la caméra répète la grande boucle (i.e., aller-retour entre les ascenseurs "New York" et "London") avant de terminer en face des ascenseurs "New York". Les numéros inscrits dans les cercles bleus indiquent les positions approximatives à partir desquelles les images de la figure 3.3 ont été acquises. Les détails sur la trajectoire de la caméra et les détections de fermeture de boucle sont donnés dans le texte.



FIG. 3.3: Images provenant du couloir du haut (première rangée) et du couloir du bas (seconde rangée) dans le plan. Ces images montrent à quel point les couloirs sont similaires. Les numéros dans les cercles bleus indiquent les positions référencées dans la figure 3.2.

Comme on peut le constater à partir de la figure 3.2, la trajectoire est colorée en bleu chaque fois que la caméra se trouve dans une zone de l'environnement qu'elle n'avait pas encore exploré, et ce en dépit de l'important aliasing perceptuel. Ce dernier est particulièrement notable dans les couloirs joignant les ascenseurs "London" et "New York", comme en attestent les images de la mosaïque donnée en figure 3.3. Au cours de cette expérience, aucun *faux positif* n'a été détecté (i.e., lorsque l'algorithme détecte une fermeture de boucle qui n'existe pas). Cela prouve la robustesse de notre solution face à l'aliasing perceptuel. D'après la figure 3.2, on peut également constater que la trajectoire apparaît en vert la plupart du temps passé dans des zones connues. Cela indique que la majorité des *vrais positifs* a été détectée (i.e., lorsqu'une fermeture de boucle est correctement détectée par l'algorithme). En particulier, la figure 3.4 donne un exemple de détection correcte.

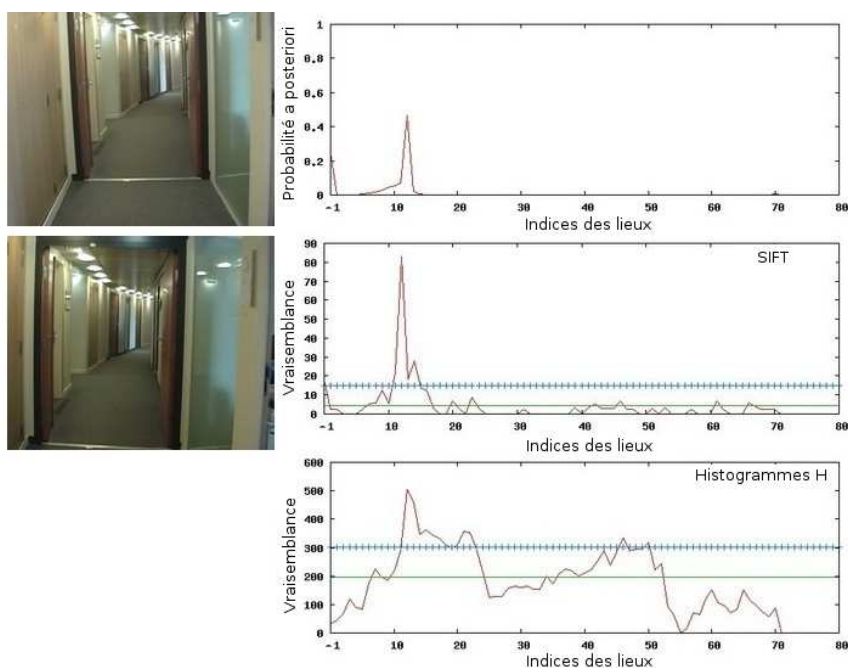


FIG. 3.4: Premier exemple de détection correcte de fermeture de boucle pour la séquence d'images d'intérieur. La distribution de probabilité a posteriori ainsi que la vraisemblance obtenue dans les espaces de représentation SIFT et histogrammes  $H$  sont présentées, de même que l'image courante  $I_t$  (haut gauche) et l'image de fermeture de boucle  $I_i$  (milieu gauche). Les vraisemblances sont obtenues à partir des scores (tf-idf) des différentes hypothèses. Pour chaque espace de représentation on donne aussi la moyenne (trait plein) et le seuil moyenne + écart type (croix bleues). Comme on peut le voir, la vraisemblance est très forte pour les images associées aux hypothèses 10 à 13, faisant en sorte que la somme des probabilités correspondantes atteigne le seuil fixé. Par ailleurs, il apparaît clairement que  $I_t$  et  $I_i$  proviennent du même lieu.

Au cours des passages de la caméra dans des zones déjà explorées, on peut noter que la couleur de la trajectoire alterne certaines fois entre rouge et vert. Cela arrive principalement aux moments correspondant



à une rotation à angle droit de la caméra autour de son axe vertical. Dans ces cas particuliers, la détection de fermeture de boucle est rejetée parce que la contrainte de géométrie épipolaire n'a pu être satisfaite : la probabilité a posteriori est au dessus du seuil fixé mais, en raison de l'importante et rapide rotation faite par la caméra, l'appariement de primitives locales nécessaire pour l'algorithme de géométrie multi-vues (voir section 2.3.3) est difficile. En effet, dans ce type d'environnement d'intérieur étroit, quand la caméra est déplacée dans un couloir tournant à angle droit, le changement de point de vue entre l'image courante et l'image de fermeture de boucle peut être important, résultant en un faible recouvrement et pénalisant au final l'association de primitives locales. Ce genre d'évènement correspond à un *faux négatif* (i.e., une fermeture de boucle qui n'est pas détectée par l'algorithme).

Lorsqu'on considère la trajectoire de la caméra en détails, on peut remarquer que la première détection de fermeture de boucle qui devrait être faite (i.e., au moment où la caméra rejoint sa position de départ pour la première fois, lors de son premier passage derrière les ascenseurs "New York") est manquée et la trajectoire apparaît toujours en bleu. Cela est dû à la faible réactivité du modèle de probabilité : la vraisemblance associée à une hypothèse en particulier doit être très importante relativement aux autres vraisemblances pour déclencher une détection de fermeture de boucle instantanée. En général, la vraisemblance associée à une hypothèse doit conserver un support significatif durant 2 ou 3 images consécutives afin de déclencher une fermeture de boucle. La réactivité de notre système est gouvernée par le modèle de transition du modèle de probabilité : on suppose ici que la probabilité de rester dans l'évènement "pas de fermeture de boucle à l'instant  $t$ " est grande (i.e., 0.9, voir section 2.3.1). Diminuer cette probabilité permettrait de détecter les fermetures de boucle plus rapidement (i.e., en requérant moins d'images), mais cela induirait également l'apparition de faux positifs dans les détections, ce qui n'est pas acceptable. Le délai nécessaire ici permet en somme d'améliorer la robustesse aux erreurs passagères de détection, considérant seulement les hypothèses ayant un support répété dans le temps comme candidats plausibles pour la fermeture de boucle.

Au cours de l'expérience, il n'y a eu qu'une seule situation pour laquelle la probabilité était au dessus du seuil alors que l'hypothèse correspondante était fausse. Cette hypothèse a donc été rejetée par l'algorithme de géométrie multi-vues. Cet évènement, que l'on considère comme une *fausse alarme*, peut être identifié dans la figure 3.2 comme la portion rouge de la trajectoire qui apparaît alors que la caméra revient pour la première fois des ascenseurs "London" (juste à côté du 6<sup>ème</sup> cercle bleu). Cette fausse alarme peut être expliquée par l'important aliasing perceptuel qui rend les couloirs joignant les ascenseurs "London" et "New York" si semblables (voir figure 3.5) : étant donné que le formalisme de sacs de mots visuels sur lequel nous nous basons (voir section 2.1) repose sur l'occurrence des mots visuels plutôt que sur leur position, l'image courante peut ressembler à un lieu du modèle sans pour autant partager avec lui une structure commune, empêchant ainsi la validation de la contrainte de géométrie épipolaire.

Afin de tester la robustesse de la détection aux changements de point de vue, nous avons changé l'orientation de la caméra en la tournant approximativement autour de son axe optique lors des deux derniers passages à l'arrière des ascenseurs "New York". Comme le montre la couleur verte de la trajectoire à ces

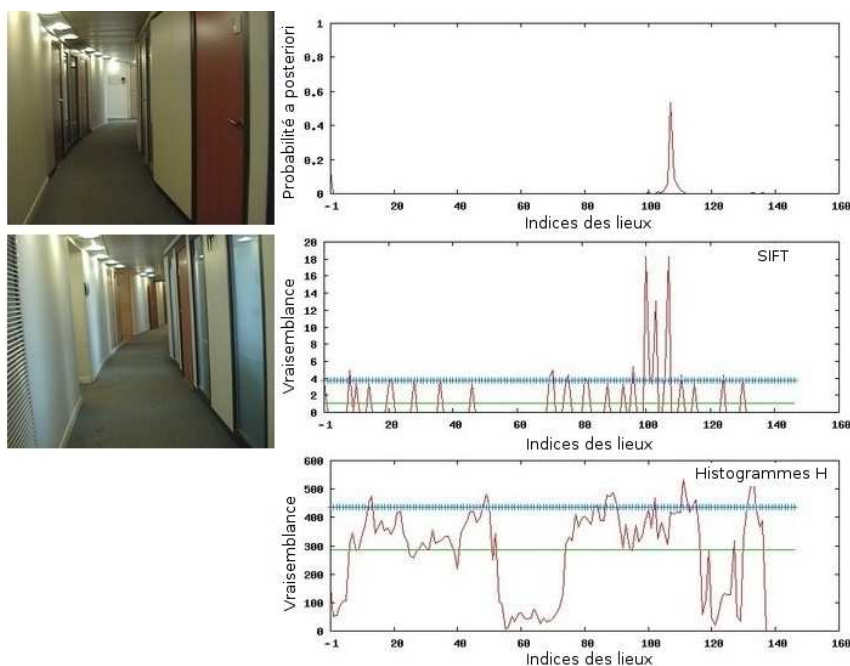


FIG. 3.5: La seule fausse alarme est due à l'aliasing perceptuel : comme on peut le constater, les vraisemblances sont confuses (on peut voir deux pics d'amplitudes comparables dans l'espace de représentation SIFT, alors que les histogrammes de teinte n'apportent pas vraiment d'information), et les images sont très similaires. Cette hypothèse a été rejetée par l'algorithme de géométrie multi-vues.

moments, les résultats de détection de fermeture de boucle n'ont pas été affectés. La figure 3.6 donne un exemple de détection de fermeture de boucle avec des orientations de caméra différentes entre l'image courante et l'image de fermeture de boucle. La détection présentée dans cette figure correspond au troisième passage de la caméra à l'arrière des ascenseurs "New York" : c'est pourquoi on observe deux pics distincts dans les vraisemblances, chacun correspondant à un des passages précédents à cet endroit.

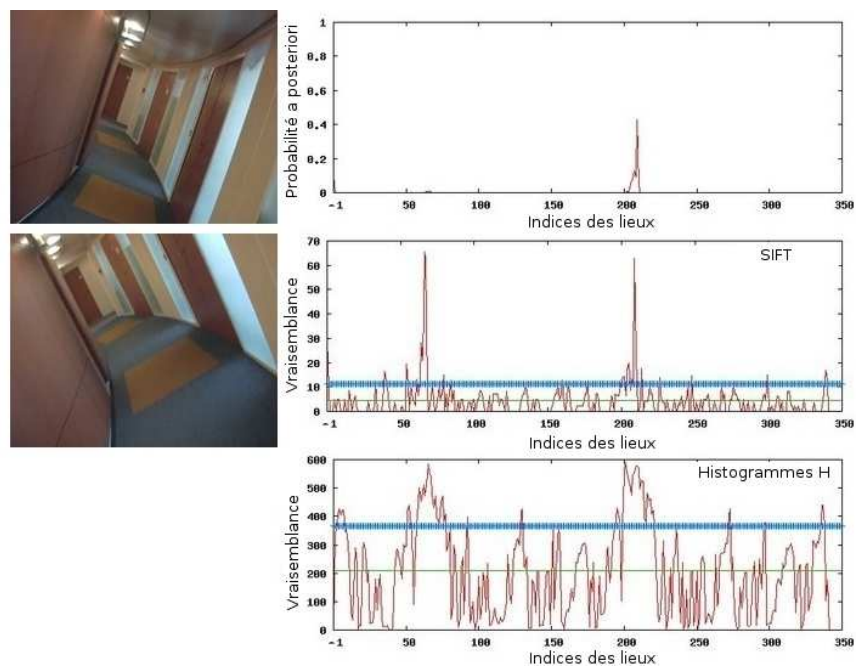


FIG. 3.6: *Second exemple de détection correcte de fermeture de boucle. Bien qu'il y ait un changement d'orientation significatif entre le point de vue du passage actuel et les points de vue des passages antérieurs, la fermeture de boucle est quand même détectée.*



FIG. 3.7: Exemples d'images provenant de la séquence d'extérieur : les images sont ordonnées par ordre d'acquisition, suivant le déplacement de la caméra lors de l'expérience.

## 3.2 Environnement d'extérieur

Pour cette seconde expérience, les images ont été acquises avec la même caméra que précédemment, mais celle-ci a cette fois été déplacée à la main dans un environnement d'extérieur, autour du bâtiment du laboratoire "Lip6" (des exemples d'images composant cette séquence sont donnés dans la figure 3.7). En raison de la plus grande profondeur de champ des scènes extérieures<sup>1</sup>, l'acquisition des images a été réalisée ici à 0.5Hz.

La trajectoire complète de la caméra à la fin de cette expérience est donnée dans la figure 3.8, en respectant le même code couleur que précédemment. Cependant, nous avons introduit ici des tirets rouge-vert pour dénoter les alternances rapides de vrais positifs et faux négatifs qui arrivent lorsque des piétons ou des voitures passent devant la caméra. Cela a pour effet nuisible de provoquer le rejet d'hypothèses à la probabilité élevée simplement parce que la contrainte de géométrie épipolaire ne peut être satisfaite. Ces événements,

<sup>1</sup>Dans l'environnement d'extérieur dans lequel a été réalisée l'expérience rapportée ici, la caméra a été déplacée en milieu urbain, en filmant dans la direction de son mouvement, le long d'avenues peu encombrées offrant des scènes avec un horizon dégagé.



dont un exemple est donné dans la figure 3.9, démontrent la robustesse du modèle de probabilité aux erreurs passagères : alors que les images sont obstruées par des piétons ou des voitures, les hypothèses correctes de fermeture de boucle sont sélectionnées (i.e., leur probabilité a posteriori est élevée), mais puisque la contrainte de géométrie épipolaire ne peut être satisfaite, elles ne peuvent pas être complètement validées pour être acceptées comme vrais positifs et sont donc rejetées.

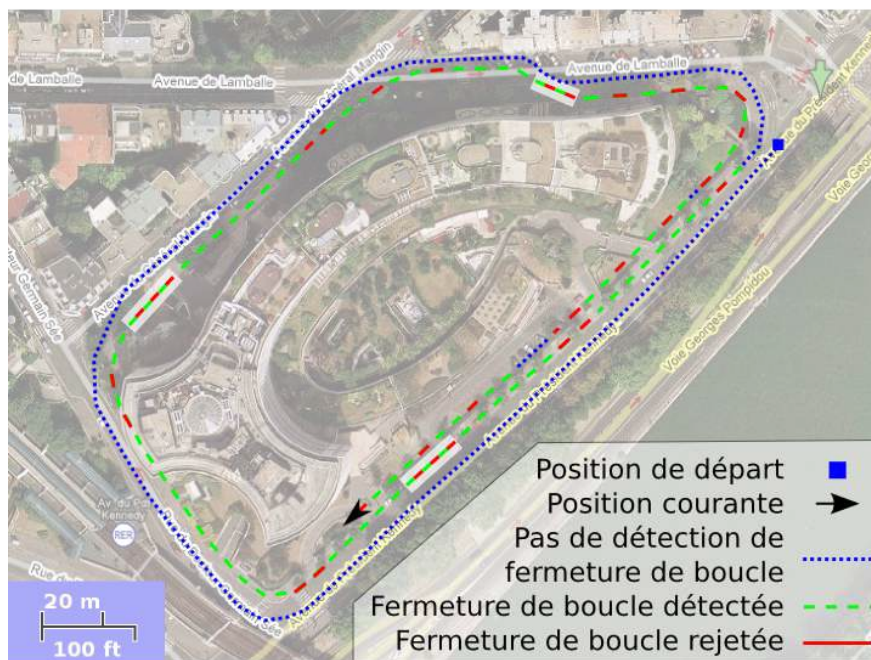


FIG. 3.8: Trajectoire de la caméra à la fin de l'expérience en extérieur. Deux boucles sont effectuées autour du bâtiment du laboratoire "Lip6", en démarrant à côté de son extrémité haut-droite (comme indiqué par le carré bleu) pour s'achever au niveau de son extrémité bas-gauche. Le chemin se trouvant en face du bâtiment (parallèle au fleuve) est donc traversé trois fois. Le code couleur employé ici est le même que celui de la figure 3.2, avec toutefois l'introduction de tirets rouge-vert qui dénotent les alternances rapides de vrais positifs et faux négatifs. Ces tirets sont superposés avec des rectangles blanc pour mieux les distinguer. Les détails sur la trajectoire et les détections de fermeture de boucle sont donnés dans le texte.

Comme dans l'expérience en intérieur, aucun faux positif n'a été trouvé, alors que plusieurs vrais positifs ont été détectés (voir figure 3.10). D'autre part, on peut voir d'après la figure 3.8 que la première détection de fermeture de boucle apparaît tardivement quand la caméra retourne sur sa position de départ pour la première fois : cela révèle à nouveau la faible réactivité du modèle de probabilité.

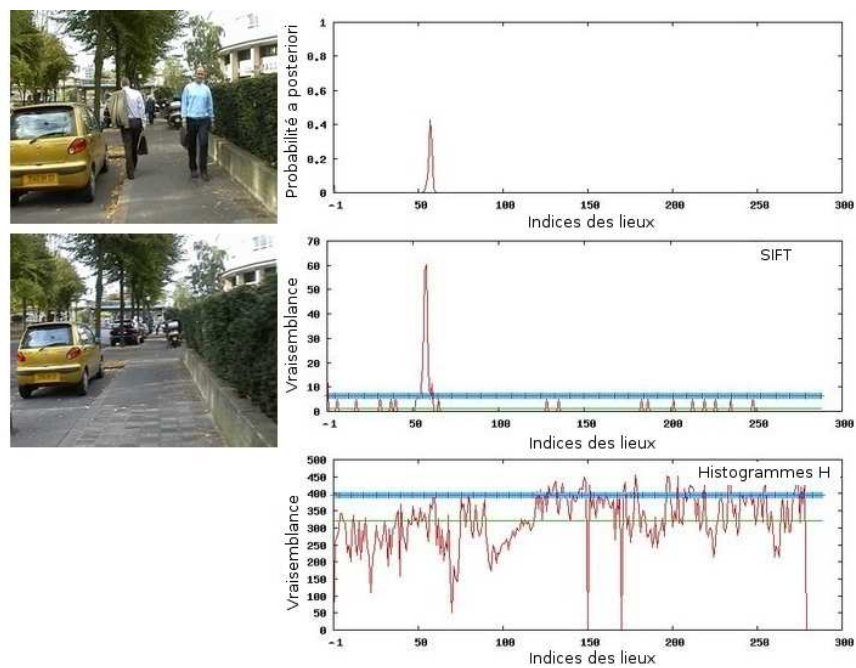


FIG. 3.9: Robustesse du modèle de probabilité aux erreurs passagères de détection : alors que l'image courante est partiellement obstruée par des piétons, une hypothèse correcte de fermeture de boucle est sélectionnée, mais elle est rejetée par l'algorithme de géométrie multi-vues.

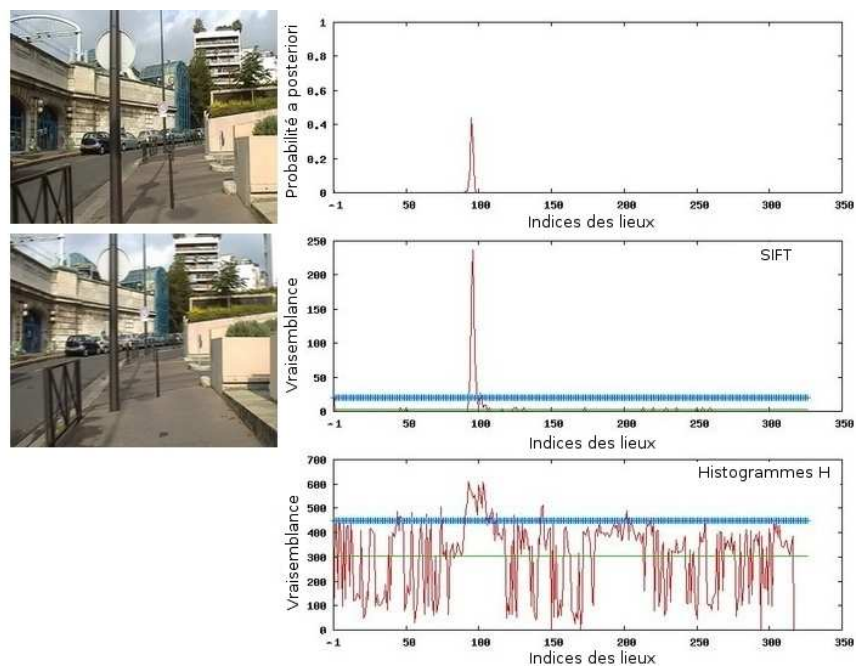


FIG. 3.10: Exemple d'une détection correcte de fermeture de boucle pour la séquence d'extérieur. Encore une fois, on peut observer que la vraisemblance obtenue dans l'espace de représentation SIFT est très importante et discriminante.

### 3.3 Analyse comparative

Dans la dernière section de ce chapitre, nous discutons les résultats expérimentaux obtenus pour la détection de fermeture de boucle. Nous commençons par une comparaison des résultats avec et sans prise en compte des différents espaces de représentation, avant d'évaluer l'impact correspondant sur les temps de calcul : la composante temps réel des traitements effectués doit rester une priorité. Nous nous intéressons également au rôle de l'algorithme de géométrie multi-vues et son importance dans le processus de détection de fermeture de boucle.

#### 3.3.1 Influence des espaces de représentation

Nous débutons cette section par une étude de l'influence des différents espaces de représentation utilisés ici (i.e., primitives SIFT et histogrammes locaux de teinte, voir section 2.1) sur la détection de fermeture de boucle. Pour cela, nous avons essayé de réaliser la détection de fermeture de boucle en utilisant uniquement les primitives SIFT ou les histogrammes H.

Nous avons vu dans les deux premières sections de ce chapitre que la combinaison de deux espaces de représentation permettait d'obtenir des résultats satisfaisants pour la détection de fermeture de boucle. Comme en attestent les résultats expérimentaux présentés dans [Angeli et al., 2008a], ainsi que l'étude comparative menée ci-après (cf. tableau 3.1), il est également possible d'atteindre des performances raisonnables de détection de fermeture de boucle en n'utilisant que les primitives SIFT pour caractériser les images.

En revanche, le taux de détection est nul lorsque les histogrammes H sont utilisés seuls pour décrire les images. Les histogrammes de teinte n'encodent qu'une information de couleur, sans prendre en compte la structure ni la texture de l'image. En conséquence, la vraisemblance associée est toujours très confuse, et elle ne sera jamais très élevée et restreinte pour une hypothèse en particulier, sauf si le lieu correspondant est identifiable par des couleurs qu'on ne trouve nulle part ailleurs. Toutefois, les histogrammes H peuvent aider à distinguer des environnements à la structure similaire et qui diffèrent uniquement du point de vue de la couleur (par exemple, deux couloirs qui ont les mêmes dimensions mais dont les murs sont peints de différentes couleurs). Lorsqu'ils sont utilisés seuls, les histogrammes de teinte ne peuvent donc pas déclencher de détection de fermeture de boucle.

Toutefois, lorsque les histogrammes H sont employés en combinaison avec les primitives SIFT, ils améliorent les détections de fermeture de boucle, permettant notamment d'obtenir une meilleure réactivité pour le modèle de probabilité. En effet, comme l'illustre la figure 3.11, la probabilité a posteriori obtenue en utilisant à la fois les primitives SIFT et les histogrammes H est souvent plus élevée que lorsque les primitives SIFT sont utilisées seules. Cela est dû au fait que les histogrammes H, même s'ils ne sont pas assez discriminants pour déclencher une détection de fermeture de boucle à eux seuls, sont légèrement plus élevés autour de l'hypothèse de fermeture de boucle, renforçant en conséquence les votes des primitives SIFT lors de la mise à jour de la probabilité a posteriori.



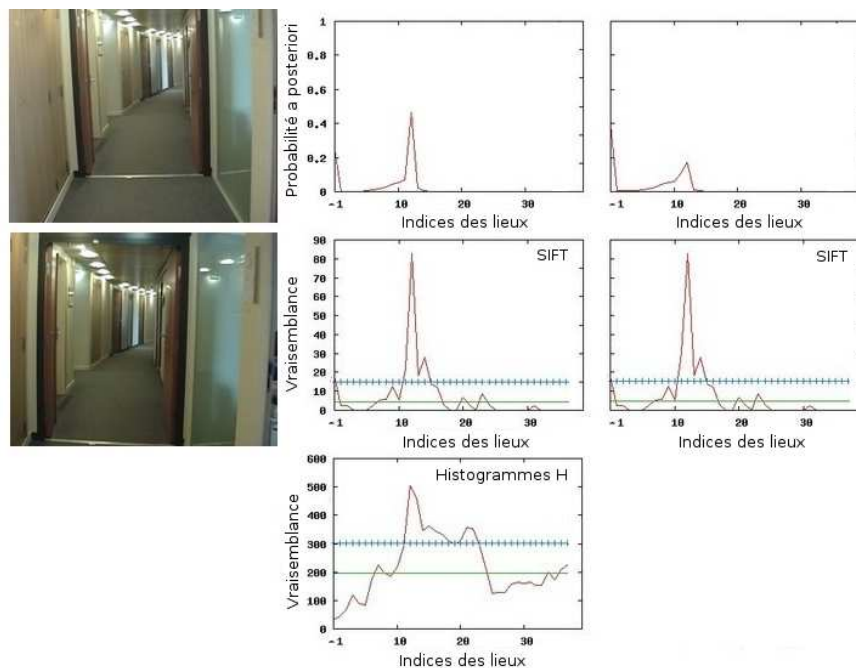


FIG. 3.11: Amélioration de la détection de fermeture de boucle par la combinaison de l'information de couleur et de texture dans la séquence d'intérieur : lorsque les espaces de représentation des primitives SIFT et des histogrammes H sont associés (partie de gauche), la probabilité a posteriori est plus élevée que dans le cas où seules les primitives SIFT sont employées (partie de droite).

L'utilisation des primitives SIFT en conjonction avec les histogrammes H permet donc d'améliorer la réactivité de l'algorithme, en permettant des détections de fermeture de boucle plus rapides, notamment lorsque la caméra revient pour la première fois sur son point de départ : les fermetures de boucles sont détectées 2 ou 3 images plus tôt lorsque les deux espaces de représentation sont pris en compte. Le tableau 3.1 donne des informations supplémentaires à propos du gain en performances dans les séquences d'intérieur et d'extérieur lorsque plusieurs représentations sont employées. Le tableau donne notamment pour chaque séquence le nombre d'images qu'elle contient (" $\#img$ "), le nombre correspondant de fermetures de boucle (" $\#FB$ ", déterminé à la main à partir de la trajectoire de la caméra), le taux de détections correspondant à des vrais positifs (" $\%VP$ ", le pourcentage de fermetures de boucles correctement détectées), et le nombre de fausses alarmes (" $\#FA$ ", hypothèses erronées qui reçoivent une forte probabilité mais qui sont rejetées par l'algorithme de géométrie multi-vues).

D'après le tableau 3.1, on peut remarquer que lorsqu'on ajoute l'information de couleur, le taux de vrais positifs est amélioré : cela est particulièrement notable dans la séquence d'intérieur où le gain est de 12%. Dans le cas de la séquence d'extérieur par ailleurs, ce gain est moins flagrant. Ceci est du à l'impressionnante robustesse des primitives SIFT dans cette séquence. En effet, étant donné que les primitives SIFT sont robustes aux changements d'échelle dans l'image, l'importante profondeur de champ des scènes extérieures

TAB. 3.1: Amélioration par l'ajout de l'information de couleur

Séquence	#img	#FB	%VP	#FA
Intérieur SIFT + H	388	217	80	1
Intérieur SIFT	388	217	68	0
Extérieur SIFT + H	531	301	71	0
Extérieur SIFT	531	301	70	0

permet une reconnaissance à long terme de ces primitives au cours du temps. Ainsi, l'ajout de l'information de couleur dans ce cas n'améliore pas les performances de manière drastique.

Bien que les tests présentés ici aient été effectués avec les deux séquences d'images, la séquence d'intérieur a produit des résultats plus intéressants, étant donné que plus de fermetures de boucles y sont effectuées au cours du trajet de la caméra, mais aussi parce que l'environnement d'intérieur est plus diversifié.

### 3.3.2 Influence de la géométrie multi-vues

D'après les résultats obtenus à partir des séquences d'intérieur et d'extérieur, nous avons remarqué qu'à plusieurs reprises certaines fermetures de boucles ont été rejetées par l'algorithme de géométrie multi-vues alors qu'elles étaient correctes : la distribution de probabilité indiquait une fermeture de boucle qui existait réellement, mais en raison des lacunes de l'algorithme de géométrie multi-vues, l'hypothèse correspondante n'a pu être définitivement entérinée. Dans le cas de la séquence d'intérieur, les rejets sont dus aux rotations rapides de la caméra lors de virages à angles droits (cf. section 3.1, paragraphe 4). Dans le cas de la séquence d'extérieur, ces rejets sont dus aux obstructions du champ de vision, en raison de la présence de piétons où de voiture en face de la caméra (cf. section 3.2, paragraphe 2).

Malgré la dégradation des performances engendrée par l'algorithme de géométrie multi-vues dans les situations indiquées ci-dessus, il reste indispensable de pouvoir se reposer sur cette ultime vérification. En effet, on peut observer d'après le tableau 3.1 que l'ajout de l'information de couleur a pour conséquence indésirable de provoquer l'apparition d'une fausse alarme : quand les primitives SIFT sont utilisées seules, aucune fausse alarme n'est détectée dans la séquence d'intérieur, alors qu'une est relevée quand les histogrammes H sont également pris en compte (cette fausse alarme est analysée dans la figure 3.5). Sans la vérification de la contrainte de géométrie épipolaire, cette fausse alarme aurait été acceptée comme un faux positif : une détection de fermeture de boucle qui n'existe pas en réalité, ce qui n'est pas acceptable.

### 3.3.3 Performances

Pendant les expériences détaillées ci-dessus, les dictionnaires des différents espaces de représentation ont été construits de manière incrémentielle, à partir d'images de taille 240x192 pixels. De plus, les temps de traitement par image ont permis d'atteindre des performances temps réel sur chacune des séquences, avec un Pentium Core2 Duo 2.33GHz. Le tableau 3.2 donne des informations au sujet de ces performances. En

particulier, ce tableau donne la longueur de chaque séquence (avec le nombre d'images correspondant), le temps CPU requis pour les traitements, et les tailles des dictionnaires obtenus à la fin de chaque expérience (cette taille est exprimée en nombre de mots). Pour chacune des séquences, nous donnons les performances lorsque les traitements ont été effectués en combinant ou non plusieurs espaces de représentation.

TAB. 3.2: *Performances*

Séquence	Longueur	#img	CPU	#SIFT	#Hist. H
Intérieur SIFT + H	6m28s	388	2m52s	9201	7284
Intérieur SIFT	6m28s	388	1m33s	9201	0
Extérieur SIFT + H	17m42s	531	10m16s	39175	18408
Extérieur SIFT	17m42s	531	6m48s	39175	0

Dans le cas de la séquence d'intérieur, les images ont été acquises à 1Hz : la caméra était déplacée dans des couloirs assez étroits et présentant une forme courbe, avec des changements de direction à angle droit apparaissant soudainement. Tout cela a motivé le choix d'une fréquence d'acquisition raisonnable de manière à ce que les images consécutives partagent des points communs. En ce qui concerne l'expérience en extérieur, les images ont été acquises à une fréquence plus faible (i.e., 0.5Hz) : les images d'extérieur acquises à des instants relativement éloignés dans le temps peuvent tout de même partager des similarités, en raison de la profondeur de champ importante des scènes d'où elles proviennent.

D'après le tableau 3.2, on observe logiquement que lorsque les primitives SIFT sont utilisées seules, le temps CPU requis pour traiter la séquence est bien plus faible que lorsque les histogrammes H sont pris en compte également : le temps total de calcul est approximativement 40% plus court dans le premier cas. Cependant, quand les deux espaces de représentation sont combinés, les traitements en temps réel sont toujours possible et, comme mentionné plus haut, la réactivité du modèle de probabilité s'en trouve améliorée, sans causer l'apparition de faux positifs. Lorsqu'une image est analysée, l'étape nécessitant le plus de ressources concerne l'extraction des primitives et les appariements avec le dictionnaire. Lorsque l'on cherche le mot du dictionnaire correspondant à une primitive extraite dans l'image courante, la recherche peut être effectuée avec une complexité logarithmique dans le nombre de mots du vocabulaire : cela est possible grâce à la structure d'arbre [Filliat, 2008] employée ici pour le dictionnaire. Sans cette structure, les performances temps réel n'auraient pu être atteintes. Afin de mieux évaluer la quantité de ressources requises par chacune des étapes de l'estimation du lieu de provenance de l'image courante, la figure 3.12 donne l'évolution des temps de calcul par image au cours du temps (pour un seul espace de représentation, en environnement d'intérieur). On peut voir que le temps nécessaire à l'extraction des primitives visuelles est presque constant, et celui-ci constitue la majeure partie des traitements. Lorsqu'on ajoute le temps de recherche des mots dans le dictionnaire, les temps de calcul évoluent de manière logarithmique. Enfin, si tous les traitements sont considérés, il semble que les temps de calcul évoluent de façon linéaire avec le temps.

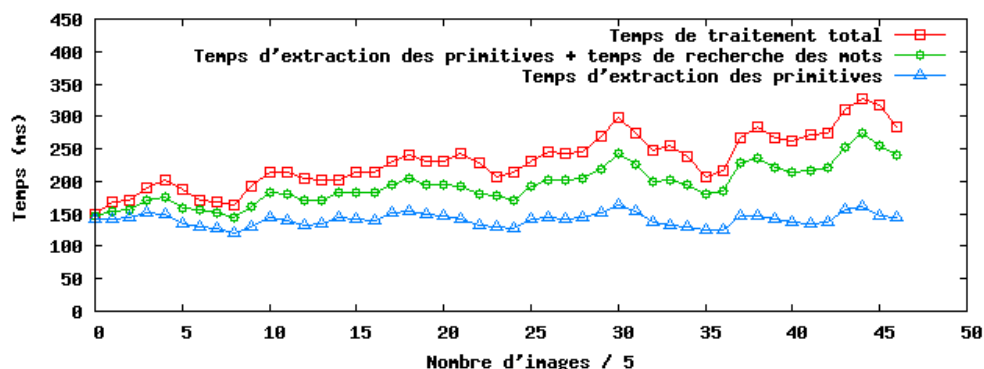


FIG. 3.12: Évolution des temps de traitement par image : la figure donne le temps requis pour extraire les primitives visuelles dans l'image (triangles), auquel est ajouté le temps nécessaire à la recherche des mots (cercles), avec également le temps de traitement total (carrés). Afin d'améliorer la lisibilité, les temps de calcul ont été moyennés toutes les cinq images.

### 3.3.4 Taille des dictionnaires

Lors de l'expérience réalisée en extérieur, la longueur de la trajectoire finale de la caméra était d'environ 1.3km, et un peu moins de 40000 mots (si on ne considère que le dictionnaire SIFT) ont été créés lors de ce déplacement, à partir de 531 images. Dans les résultats obtenus par les auteurs de [Cummins and Newman, 2007] concernant la détection de fermeture de boucle dans un cadre proche des travaux présentés ici, la collection d'information pour la construction hors-ligne du dictionnaire a été faite sur 30km, analysant 3000 images pour générer au final 35000 mots.

En comparaison, il apparaît donc clairement que notre modèle requiert un nombre beaucoup plus élevé de mots que dans la solution proposée par [Cummins and Newman, 2007]. L'explication intuitive de cette différence tient en deux points. Premièrement, dans le cadre incrémentiel de la construction du dictionnaire, on ne peut se permettre de réordonner l'information collectée, ce qui pourrait rendre la représentation correspondante plus compacte. Deuxièmement, afin que les coefficients  $tf-idf$  employés ici pour le calcul de la vraisemblance soient efficaces et permettent de distinguer aisément les hypothèses, il faut que le dictionnaire contienne des mots discriminants. Ainsi que le montrent les résultats obtenus dans [Filliat, 2007], le rayon des mots (i.e., la taille des boules caractérisant les mots dans l'espace de leurs descripteurs) a une influence directe sur le pouvoir discriminant du modèle de vraisemblance : on peut élever ce pouvoir en diminuant le rayon, mais cela résulte en un dictionnaire de grande taille (i.e., contenant un nombre important de mots).

Les paramètres utilisés ici ont été déterminés de manière empirique et ont permis d'obtenir des résultats satisfaisants sur tous les types d'environnement rencontrés.

# Chapitre 4

## Discussion

### 4.1 Apprentissage du dictionnaire

La solution que nous proposons pour la détection de fermeture de boucle est complètement incrémentielle et elle permet des traitements en temps réel sur la base d'une caméra monoculaire uniquement. Le formalisme incrémentiel des sacs de mots visuels développé par l'auteur de [Filliat, 2007] et employé ici permet une gestion efficace et uniforme de plusieurs caractérisations d'image, dans des espaces de représentation différents. L'hétérogénéité de l'information visuelle utilisée pour l'estimation permet notamment d'améliorer les performances de classification d'image pour la tâche considérée. En effet, il apparaît que la combinaison des primitives SIFT (renseignant sur la structure locale dans l'image), et des histogrammes locaux de teinte (prenant en compte la couleur), augmente la réactivité du modèle de probabilité tout en respectant la contrainte de temps réel. Par ailleurs, la construction incrémentielle du dictionnaire offre la possibilité de n'apprendre que la partie de l'environnement dans lequel évolue le robot. Cela constitue une différence majeure face aux approches citées dans l'état de l'art de cette partie et qui encodent aussi les images selon le paradigme des sacs de mots visuels (i.e., [Fraundorfer et al., 2007], [Cummins and Newman, 2007], [Newman et al., 2006], [Wang et al., 2006]) : dans ces travaux, le dictionnaire est appris lors d'une phase préalable hors-ligne au cours de laquelle on analyse une importante base de données d'images collectées dans des lieux supposés représenter le type d'environnement dans lequel le robot se trouvera par la suite. Notre système fait donc preuve d'une plus grande adaptabilité, permettant la détection de fermeture de boucle dans des environnements d'intérieur et d'extérieur (voire même dans des environnements mixtes, comme nous le verrons plus tard), sans avoir à sélectionner à l'avance le type d'environnement à apprendre.

Cependant, cette approche incrémentielle présente l'inconvénient de débiter l'expérience avec un dictionnaire vide, ce qui peut être problématique. Notre système repose sur un mécanisme de lieux virtuels pour représenter l'évènement "pas de fermeture de boucle à l'instant  $t$ " et pour en évaluer la vraisemblance à chaque nouvelle image traitée. Nous avons déjà insisté sur l'importance de cet évènement (voir section

2.3.2), car c'est lui qui est en partie responsable de la gestion de l'aliasing perceptuel : en lui associant un score élevé dans les situations ambiguës, on évite que des hypothèses erronées reçoivent une probabilité trop importante. La construction du lieu virtuel est réalisée à partir de statistiques sur l'occurrence des mots dans les lieux connus. Or, en début d'expérience, lorsque le vocabulaire contient peu de mots vus peu de fois, il est difficile d'extraire des statistiques pertinentes pour la construction du lieu virtuel : il arrive de ce fait que le score de ce lieu mette un certain temps (une dizaine d'images) pour atteindre une valeur confortable empêchant toute hypothèse erronée de recevoir une probabilité élevée. Dans les séquences considérées plus haut, cela n'a pas été un problème, mais dans d'autres environnements, une fausse alarme pourrait être relevée.

L'approche retenue pour l'apprentissage du vocabulaire est donc cruciale pour la tâche abordée. D'une part, les méthodes basées sur l'apprentissage préalable d'un dictionnaire statique posent la question de la généralisation : à quel point les mots appris sur des images en ville peuvent être utiles à la détection de fermeture de boucle dans un environnement d'intérieur, où dans le cadre d'une exploration sous-marine ? Est-il possible d'apprendre un vocabulaire qui soit pertinent dans toutes les situations tout en offrant une représentation compacte et à la mesure des expériences réalisées ? D'autre part, les méthodes incrémentielles posent des problèmes de conditions initiales : comment raisonner sur les statistiques des mots lorsque ceux-ci sont distribués uniformément ? L'idéal serait probablement une solution intermédiaire, partant d'un modèle générique compact qui pourrait être mis à jour avec de nouvelles informations, ou bien même échangé contre un autre s'il n'était pas pertinent.

## 4.2 Gestion de l'aliasing perceptuel

Les résultats de détection de fermeture de boucle présentés ici démontrent la robustesse de notre solution en présence d'aliasing perceptuel dans l'environnement. Cependant, le modèle de probabilité plus complexe développé par les auteurs de [Cummins and Newman, 2007] permet de mieux gérer ce phénomène. En effet, dans cette approche l'aliasing perceptuel est pris en compte au niveau de l'information visuelle élémentaire de base pour l'estimation (i.e., les mots visuels). Pour cela, lors de la construction hors-ligne du dictionnaire, les probabilités de co-occurrence des mots sont approximées. Ensuite, lors de l'estimation du lieu de provenance de l'image courante, ces probabilités de co-occurrence servent à définir le modèle de vraisemblance. Ainsi, ce modèle n'évalue pas la pertinence des mots un à un, comme dans le modèle que nous avons développé, mais il évalue au contraire la pertinence de l'apparition simultanée d'un ensemble de mots dans un même lieu. Le modèle appris synthétise donc fidèlement les caractéristiques de l'environnement sur la base de l'occurrence simultanée des mots, et il assure une discrimination efficace des lieux semblables dans le cadre probabiliste global. Dans ce cadre, les hypothèses qui sont des erreurs de détection temporaires reçoivent un support faible, étant donné que du fait de leur caractère temporaire, elles ne correspondent certainement pas à une co-occurrence probable de mots.

Dans notre système, la prise en compte de l'aliasing perceptuel est réalisée à la suite de l'estimation de la probabilité de fermeture de boucle, grâce à l'algorithme de géométrie multi-vues : celui-ci permet d'écarter des hypothèses semblables mais ne partageant pas de structure commune. Cette ultime étape de vérification permet de combler les lacunes du modèle de vraisemblance, en rejetant des hypothèses erronées (une seule fausse alarme a du être écartée en tout et pour tout, voir figure 3.5). Cela a également l'inconvénient de rejeter certaines hypothèses correctes, en cas de rotations rapides de la caméra sur son axe vertical ou d'obstructions partielles du champs de vision (voir section 3.1, paragraphe 4, ou encore section 3.2, paragraphe 2). Ainsi, un modèle de probabilité plus robuste permettrait d'obtenir des résultats équivalents voire supérieurs tout en se passant de l'étape de vérification de la contrainte de géométrie épipolaire. Toutefois, l'information de transformation relative entre l'image courante et le lieu de fermeture de boucle est indispensable à la correction de la position et de l'orientation de la caméra dans le cadre de l'application au SLAM métrique, comme nous le verrons plus loin. D'autre part, cette information peut également servir d'odométrie visuelle dans le cadre du SLAM topologique, pour exprimer les contraintes relatives entre noeuds voisins.

En définitive, même s'il existe des modèles de probabilité ne nécessitant pas de validation par géométrie multi-vues pour la détection de fermeture de boucle ([Cummins and Newman, 2007]), l'information apportée se révèle utile voire indispensable pour les applications de SLAM associées. De plus, la pertinence du modèle de probabilité proposé par les auteurs de [Cummins and Newman, 2007] provient essentiellement de l'apprentissage des probabilités de co-occurrence des mots. Or, cet apprentissage est réalisé au cours d'une phase préalable hors-ligne, rendant de ce fait une implémentation totalement incrémentielle impossible en l'état.

### 4.3 Caractéristiques du modèle de probabilité

Dans la section relative aux résultats expérimentaux de cette partie, il a été remarqué à plusieurs reprises que notre modèle de probabilité manquait de réactivité : une fermeture de boucle est détectée jusqu'à cinq images après son apparition effective. Ceci est imputable aux paramètres du modèle de transition (voir section 2.3.1), comme expliqué précédemment (voir section 3.1, paragraphe 5), et peut potentiellement être problématique lorsque la caméra repasse plusieurs fois au même endroit : le délai existant entre une fermeture de boucle et sa détection provoque l'ajout de nouvelles entrées dans le modèle correspondant en fait à des lieux connus. En conséquence, plusieurs hypothèses identifiant les passages antérieurs sont plausibles et reçoivent un score élevé lorsque la caméra retourne sur une position visitée plusieurs fois par le passée (voir figure 4.1). Dans ces conditions, il devient difficile de sélectionner une hypothèse de manière univoque. Heureusement, l'ambiguïté est naturellement levée après avoir traité quelques images.

Le modèle de probabilité développé ici et la représentation de l'environnement associée offrent une complexité globale linéaire en le nombre de lieux recensés. Le calcul de la vraisemblance repose sur un index inversé pour éviter une comparaison exhaustive de l'image courante avec tous les lieux connus. Ce

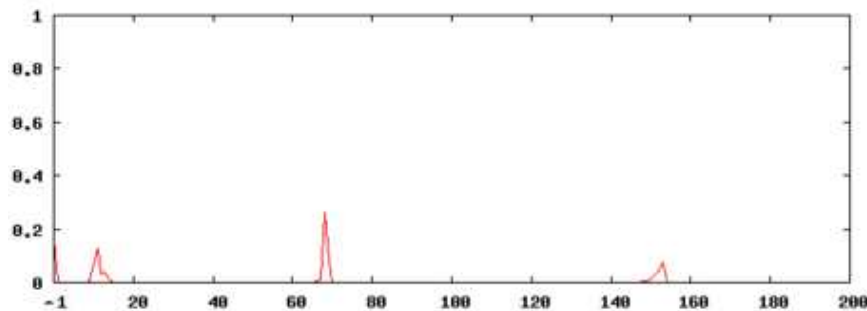


FIG. 4.1: Exemple de distribution de probabilité a posteriori ambiguë : l'hypothèse virtuelle (indice  $-1$ ) mise à part, on aperçoit trois pics d'amplitude non négligeable qui correspondent aux trois différents passages au même endroit.

mécanisme simple assure une mise en oeuvre rapide pour l'estimation de la probabilité de fermeture de boucle. D'autre part, la méthode RANSAC employée pour la vérification de la contrainte de géométrie épipolaire rend cette ultime étape de validation des hypothèses réalisable en un temps limité. Enfin, grâce à la complexité logarithmique en le nombre de mots des opérations de recherche dans le dictionnaire, la comparaison d'une primitive visuelle extraite dans l'image courante avec le vocabulaire correspondant peut être faite rapidement. Ce sont ces caractéristiques de l'implémentation qui ont permis au final d'obtenir la réalisation des traitements en temps réel sur toutes les séquences testées.

## 4.4 Conclusion

La solution proposée pour la détection de fermeture de boucle satisfait les contraintes énoncées dans l'introduction de ce mémoire. La méthode mise en oeuvre est indépendante de tout algorithme de SLAM, et elle repose sur son propre modèle de l'environnement. Celui-ci est construit de manière incrémentielle et ne requiert aucune phase d'apprentissage préalable hors-ligne. Tous les traitements nécessaires à la détection de fermeture de boucle ont pu être réalisés en temps réel sur les séquences d'images testées. De plus, la technique employée est basée sur un cadre probabiliste, ce qui permet de mesurer le niveau de confiance associé à l'estimation. L'approche proposée permet par ailleurs de prendre en compte différentes caractérisations des images, dans des espaces de représentation hétérogènes, et ce de manière uniforme quelque soit le type de primitive locale considérée. Dans la suite de ce mémoire, nous allons voir de quelle manière cette solution peut être adaptée à des applications de SLAM topologique et métrique.



## **Deuxième partie**

# **Application au SLAM topologique**



## Chapitre 5

# État de l'art

Dans la première partie de ce mémoire, nous avons détaillé notre solution de détection de fermeture de boucle basée sur l'apparence uniquement. Nous proposons dans cette deuxième partie une application directe de ces travaux à la construction d'une carte topologique de l'environnement. Cependant, nous restons toujours dans le contexte d'une approche complètement incrémentielle, et où les traitements doivent être réalisables en temps réel. Ainsi, la construction de la carte doit être faite au fur et à mesure que les images sont acquises par le robot, et non au cours d'un processus hors-ligne qui comparerait un ensemble d'images représentatives de l'environnement et acquises sur toutes les zones dans lesquelles le robot évoluera par la suite (toutes les pièces d'un bâtiment par exemple).

Par conséquent, le problème considéré est le SLAM topologique : chaque fois qu'une nouvelle image est acquise, il faut décider de son lieu de provenance. A partir du résultat de cette décision, le modèle de l'environnement doit être mis à jour. Dans le cas où l'image courante décrit un nouveau lieu, celui-ci doit être ajouté. Si cette image provient d'un lieu existant, il faut mettre à jour la description de ce dernier. Déterminer le lieu de provenance de l'image courante dans une carte constitue la partie *localisation* du problème du SLAM, alors que la décision résultante concerne la partie *cartographie*.

L'état de l'art réalisé dans ce chapitre ne concerne que les approches au problème du SLAM topologique qui reposent sur une information qualitative, et sur la vision en particulier. Ainsi, nous ne considérons pas les méthodes qui prennent en compte d'autre type d'information pour l'estimation. Par exemple, les approches basées explicitement sur l'information métrique fournie par l'odométrie d'un robot ou par une technique d'odométrie visuelle pour la construction de la carte ne seront pas présentées. Cependant, une brève revue des méthodes de SLAM topologique est réalisée en introduction de cette section. Le reste de cet état de l'art est ensuite scindé en deux sections, afin de distinguer les méthodes reposant sur la vision uniquement de celles plus généralistes qui peuvent employer d'autres modalités sensorielles, ou même se baser sur la séquence d'actions effectués par le robot. Dans le premier type d'approche, le modèle d'observation occupe une place prépondérante, et la qualité de la vraisemblance qu'il permet d'estimer est primordiale pour le

reste du processus d'inférence. Dans le second type d'approche au contraire, les modèles d'observations peuvent être simplistes, et la viabilité de l'inférence est assurée par d'autres moyens, comme la cohérence entre les séquences d'observations faites lors du trajet du robot et les séquences de noeuds traversés dans la carte.

## 5.1 Revue générale

Depuis les premières activités de recherche sur la localisation et la cartographie, dans les années 80, deux formalismes ont émergé pour la représentation de l'environnement. En effet, comme nous l'avons déjà mentionné dans l'introduction de ce mémoire, dans les applications de SLAM l'environnement peut être modélisé de façon métrique ou topologique. La distinction entre ces deux types d'approches n'est pourtant pas claire : les cartes topologiques peuvent inclure une information métrique de position pour les noeuds qu'elles contiennent, sous la forme de coordonnées cartésiennes précises, alors que les cartes métriques peuvent être hiérarchisées dans un graphe de plus haut niveau. C'est en fin de compte la granularité de l'information enregistrée dans la carte qui permettra de mieux distinguer ces deux familles de représentation, les approches topologiques offrant notamment un niveau de modélisation plus symbolique.

Parmi les plus anciens travaux abordant le problème du SLAM, la solution exposée dans [Kuipers and Byun, 1991] est certainement une des premières à proposer un formalisme topologique précis pour la représentation de l'environnement. D'une manière générale, d'après ce formalisme, l'environnement est segmenté en lieux décrits par des noeuds et connectés entre eux par des arêtes, formant ainsi un graphe. Ainsi, les méthodes abordant le problème du SLAM topologique diffèrent principalement dans leur façon de définir les noeuds et les arêtes.

### Définition des noeuds

Le choix de l'information enregistrée dans les noeuds de la carte dépend évidemment des capacités des capteurs embarqués sur le robot, et c'est généralement l'apparence qui sert à décrire les lieux : un scan laser [Lu and Milios, 1997], une image omnidirectionnelle [Booiij et al., 2007], ou encore une collection de primitives locales extraites dans différentes vues proches [Cummins and Newman, 2007] constituent quelques exemples du type d'information qui peut être prise en compte. On peut également distinguer les différentes approches par le mécanisme de création des lieux qu'elles mettent en oeuvre. Celui-ci peut être complètement supervisé : les auteurs de [Booiij et al., 2007] créent systématiquement un nouveau noeud chaque fois qu'une image est acquise. Dans certaines solutions, la création d'un noeud est déclenchée lors d'observations canoniques ([Simmons and Koeing, 1995]). Ainsi, dans ces méthodes, le modèle d'observation ne permet de distinguer qu'un nombre limité de types de lieux (couloirs, passages de portes...). Enfin, il existe des solutions où la création des noeuds est faite de manière totalement non-supervisée, en comparant

chaque nouvelle image acquise avec les lieux du modèle de l'environnement construit jusque-là ([Cummins and Newman, 2007]).

### Définition des arêtes

Les arêtes servent à encoder les relations existant entre les différents noeuds. Dans le plus simple des cas, c'est la relation d'adjacence entre les noeuds qui est enregistrée, indiquant quels lieux sont atteignables depuis un noeud donné ([Ulrich and Nourbakhsh, 2000]). Cette simple relation d'adjacence peut être augmentée avec une information indiquant le chemin à suivre pour naviguer localement entre les noeuds. La route ainsi enregistrée peut être donnée par une information métrique obtenue par odométrie ([Hubner and Mallot, 2007]) ou odométrie visuelle ([Konolige and Agrawal, 2007]).

### Caractéristiques générales

La nature de l'information utilisée pour la caractérisation de l'environnement sous la forme d'une carte topologique permet une mise en correspondance simple entre les mesures provenant des capteurs et les noeuds de la carte. En effet, la description des lieux utilisée offre l'avantage d'être plus directement liée aux capacités perceptives du robot. D'autre part, dans ce genre de représentation, et lorsqu'une information métrique est encapsulée dans la carte, les perceptions du robot (i.e., les mesures effectuées grâce aux capteurs) sont distinctement séparées de ses proprioceptions (i.e., l'odométrie). Ainsi, les sources de bruit relatives à ces deux types d'information restent bien séparées et leurs influences respectives ne sont pas mélangées. Par ailleurs, le niveau symbolique de représentation des cartes topologiques simplifie la planification de trajectoire, celle-ci étant traitée comme une recherche de plus court chemin dans un graphe. Toutefois, en représentant l'environnement par un tel graphe, il est impossible de planifier précisément un déplacement dans une zone encore non visitée : pour être complète, la carte nécessitera une exploration exhaustive de l'environnement.

Le lecteur intéressé pourra trouver une présentation plus détaillée et un historique plus complet au sujet des approches topologiques dans [Filliat, 2001], [Filliat and Meyer, 2003] ou encore [Thrun, 2002].

## 5.2 Méthodes basées sur la vision uniquement

La plupart des approches basées sur l'apparence développées dans le cadre du SLAM ou de la construction de carte topologiques reposent sur la vision omnidirectionnelle ([Booi et al., 2007], [Goedemé et al., 2007], [Valgren et al., 2007], [Zivkovic et al., 2005]). Ces méthodes ayant déjà été abordées dans l'état de l'art de la partie I, elles ne seront donc que brièvement résumées ici. Dans ces approches, une mesure de similarité entre images est employée pour définir les noeuds ou les arêtes de la carte. Par exemple, dans les travaux de [Booi et al., 2007], chaque image acquise pour la construction de la carte sert à identifier un lieu

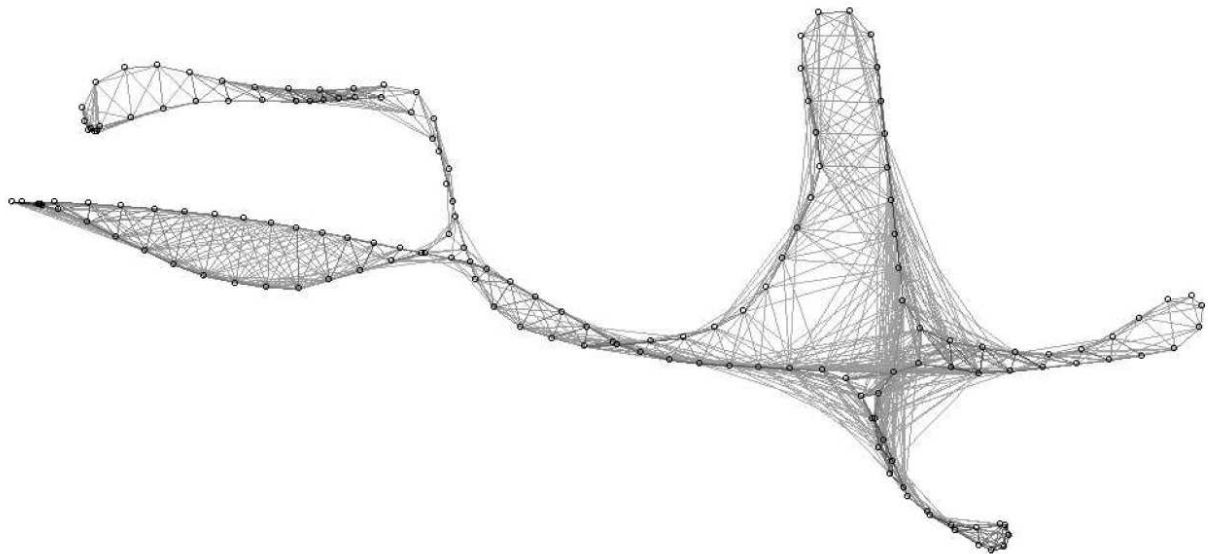


FIG. 5.1: Exemple de carte topologique générée à partir d'une caméra omnidirectionnelle ([Booiij et al., 2007]). Les noeuds indiquent les différentes positions où les images ont été acquises, alors que les arêtes lient les images similaires. On remarque que grâce à la vision omnidirectionnelle, deux points de vue relativement distants peuvent tout de même partager un certain nombre de similarités.

(i.e., un noeud), alors que la mesure de similarité permet de lier les lieux entre eux sur la base de leurs points communs (i.e., plus les lieux sont semblables, plus leur lien est fort, voir figure 5.1). Dans une perspective identique, les auteurs de [Zivkovic et al., 2005] construisent une matrice de similarité sur la base de la comparaison des images : les éléments hors diagonaux non nuls de cette matrice servent à définir les arêtes de la carte (i.e., les images désignées par ce genre d'élément dans la matrice correspondent à des noeuds similaires dans la carte). Enfin, dans la solution proposée par [Goedemé et al., 2007], différentes mesures de similarité entre images sont combinées pour définir des groupes d'images correspondant à des lieux semblables et identifiant les noeuds de la carte. Les groupes peuvent alors être fusionnés grâce à une méthode dérivée de la théorie de l'évidence pour effectuer des fermetures de boucles dans la carte construite. Dans ces trois méthodes, la construction de carte est réalisée lors d'une phase hors-ligne à partir d'images représentatives de l'environnement. L'approche mise en oeuvre dans les travaux de [Valgren et al., 2007], et dont un exemple de résultat est donné dans la figure 5.2, concerne quant à elle la problématique du SLAM topologique. La mesure de similarité entre images sert à déterminer le lieu de provenance de l'image courante, afin de mettre à jour la carte en conséquence. En particulier, l'algorithme *incremental spectral clustering*, reposant sur la construction en-ligne d'une matrice de similarité, permet de déterminer le lieu de l'image courante.

Les approches basées sur l'apparence et reposant sur la vision omnidirectionnelle permettent une segmentation efficace de l'environnement : les images omnidirectionnelles offrent la possibilité de reconnaître un lieu depuis des points de vue distants. Cependant, aucune des méthodes mentionnées ci-dessus ne satis-

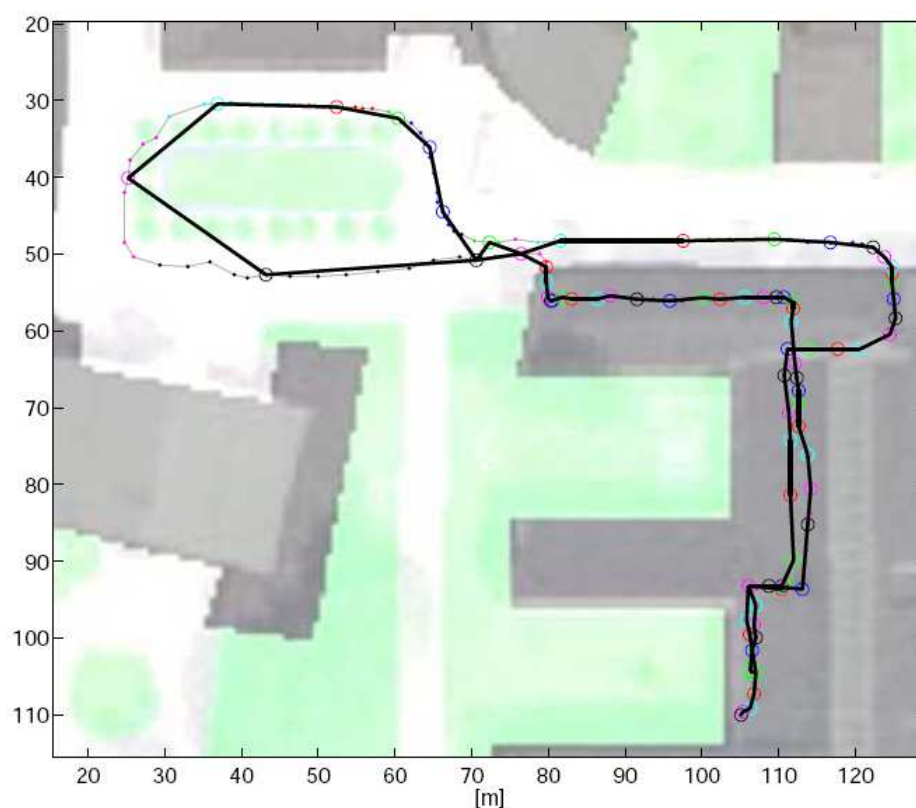


FIG. 5.2: Second exemple de carte topologique générée à partir d'une caméra omnidirectionnelle ([Valgren et al., 2007]). Les points indiquent les positions où les images ont été acquises, et chacun d'entre eux est associé à un noeud de la carte représenté par un cercle de la même couleur. Les arêtes (trait épais) lient les noeuds en fonction de leur ordre de parcours.

fait à la fois les contraintes de traitements incrémentiels et en temps réel : soit la carte est construite lors d'un processus préalable hors-ligne ([Booi et al., 2007], [Goedemé et al., 2007], [Zivkovic et al., 2005]), ou alors la complexité de la détermination du lieu de l'image courante empêche sa réalisation en temps réel ([Valgren et al., 2007]).

Les auteurs de [Fraundorfer et al., 2007] présentent une solution pour le SLAM topologique basée sur la vision monoculaire et dont les traitements sont réalisés en temps réel. Dans cette méthode, les images sont encodées selon le paradigme des sacs de mots visuels (voir section 1.4.1) : chaque image est représentée par un vecteur contenant une entrée par mot du dictionnaire qui renseigne sur la présence de ce mot dans l'image. Ainsi, les vecteurs identifiant les images ont une taille égale au nombre de mots dans le vocabulaire. Ce dernier est par ailleurs construit lors d'une phase préalable hors-ligne sur la base d'images représentatives de l'environnement. Pour déterminer le lieu de l'image courante, celle-ci est comparée à toutes les images traitées jusque-là dans le cadre d'une méthode de vote avec validation par un algorithme de géométrie multi-vues : les  $n$  images recevant le plus de votes sont analysées grâce à l'algorithme de géométrie multi-vues

afin de ne sélectionner que celle qui ressemble à l'image courante et qui partage une structure commune avec elle. Lors du vote, les images sont comparées en calculant la norme L2 séparant les vecteurs associés. Cette approche ressemble donc en de nombreux points à la solution présentée dans le cadre de ce mémoire. Cependant, l'implémentation retenue pour la construction du dictionnaire n'est pas incrémentielle et repose sur un processus hors-ligne préalable à la phase d'exploitation.

Dans un contexte proche, les auteurs de [Cummins and Newman, 2007] proposent un algorithme de SLAM topologique basé sur l'apparence uniquement et reposant sur une simple caméra monoculaire. Comme cela a déjà été brièvement décrit dans le chapitre 1 de la partie I (section 1.3.3), la solution exposée dans ces travaux implémente un modèle de probabilité robuste à l'aliasing perceptuel pour déterminer le lieu de provenance de l'image courante. Pour cela, les images sont caractérisées selon le formalisme des sacs de mots visuels, et un modèle génératif de l'apparence de l'environnement est appris au cours d'une phase préalable hors-ligne nécessitant le traitement de plusieurs milliers d'images. Au cours de cet apprentissage, le dictionnaire servant ensuite à encoder les images est construit, alors que les probabilités de co-occurrence des mots le constituant sont estimées pour définir le modèle génératif de l'apparence de l'environnement. L'algorithme de SLAM topologique mis en oeuvre caractérise un lieu par un ensemble de vues proches avec une complexité linéaire en le nombre de lieux. Cependant, malgré les améliorations apportées dans [Cummins and Newman, 2008b] et destinées à accélérer les traitements, il n'est pas encore possible d'atteindre des performances en temps réel, et l'implémentation générale n'est pas incrémentielle puisqu'une phase hors-ligne d'apprentissage est requise.

### 5.3 Autres modalités perceptives

Dans les travaux de [Simmons and Koeing, 1995], les auteurs proposent d'employer le formalisme des Processus de Décision Markoviens Partiellement Observables (PDMPO) afin d'aborder le problème du SLAM topologique. Dans les modèles de PDMPO, l'environnement est considéré comme un ensemble d'instances de classes de lieux (i.e., couloir, pièce, passage de porte) formant les noeuds de la carte. Ainsi, en fonction de l'apparence de la zone de l'environnement qu'il représente, un noeud sera associé à une classe déterminant ses caractéristiques (i.e., un couloir est identifiable par sa largeur, un passage de porte par son étroitesse et une pièce par sa superficie). L'apparence d'une classe permet de définir un modèle d'observation pour chaque type de noeud. Dans les PDMPO on définit également un modèle de transition liant les différents types de lieu (i.e., un passage de porte fait le lien entre une pièce et un couloir, il est peu probable de passer d'un couloir à une pièce directement). A partir de ces modèles d'observation et de transition, la position du robot dans l'environnement est estimée sous la forme d'une distribution de probabilité sur les noeuds : le modèle d'observation prédit la classe la plus plausible pour le lieu courant compte tenu de son apparence, alors que le modèle de transition permet de propager l'estimation dans le temps. Récemment, les auteurs de [Tomatis et al., 2003] ont amélioré ce formalisme en permettant la



détection de fermeture de boucle sur la base d'un laser à  $360^\circ$  : lorsque deux séquences distinctes de noeuds adjacents comptabilisent à elles seules et en quantités comparables tout la masse de probabilité, alors ces noeuds représentent certainement les mêmes lieux de l'environnement, et ils peuvent donc être fusionnés. Dans les travaux de [Tapus and Siegwart, 2005], un laser à  $360^\circ$  est également employé, en combinaison avec une caméra omnidirectionnelle, pour caractériser les noeuds du PDMPO par les *empreintes visuelles* des lieux correspondants (voir figure 5.3). Cependant, le problème de la détection de fermeture de boucle n'est pas abordé. Les modèles de PDMPO ne sont généralement pas adaptés à la construction de carte en ligne, puisqu'ils doivent être appris lors d'une phase préalable hors-ligne ([Shatkay and Kaelbling, 1997]), ou bien paramétrés manuellement à partir d'informations a priori sur l'apparence de l'environnement, sa géométrie et sa topologie (i.e., l'environnement ne contient que des couloirs et des pièces, les couloirs ont tous la même largeur et ils se croisent à angles droits, le robot est supposé se trouver dans un couloir la plupart du temps).

Des méthodes d'inférence Bayésienne ont été investiguées par les auteurs de [Savelli and Kuipers, 2004] et de [Ranganathan et al., 2006] afin d'apprendre une carte de l'environnement. L'idée consiste à faire des tirages aléatoires d'échantillons dans l'espace des topologies de manière à générer des cartes dont les caractéristiques sont ensuite mises en correspondance avec des informations obtenues à partir du robot. Ainsi, les cartes générées qui sont vraisemblables compte tenu des mesures provenant des capteurs [Ranganathan et al., 2006] (cf. figure 5.4) ou bien des actions effectuées par le robot [Savelli and Kuipers, 2004] sont sélectionnées comme étant les cartes les plus probables. A partir des topologies ainsi obtenues, de nouveaux tirages sont effectués pour trouver des échantillons encore plus probables sur la base des dernières informations relevées. En particulier, les auteurs de [Savelli and Kuipers, 2004] définissent des contraintes pour améliorer la cohérence des cartes inférées (la "planarité" sert par exemple à imposer que la carte soit contenue dans le plan et que ses arêtes ne se croisent pas). Ces approches sont incrémentielles, et elles permettent d'apprendre des cartes valides même en cas de multiples fermetures de boucles, à partir de mesures capteur simplistes ou seulement sur la base de la séquence d'actions effectuées par le robot. Toutefois, la complexité des méthodes d'inférence sur lesquelles elles reposent en limitent l'application à des environnements simples, contenant peu de lieux distincts (i.e., jusqu'à une quinzaine de noeuds dans la carte).

## 5.4 Conclusion

De manière similaire à ce que nous avons observé dans la partie I au sujet de la détection de fermeture de boucle, il semble que les solutions présentées ici peinent à combiner mise en oeuvre complètement incrémentielle et traitements réalisables en temps réel. Certaines approches proposent des solution incrémentielles pour l'inférence qui reposent sur des modèles d'observation basiques, mais la complexité du processus d'estimation sous-jacent les empêche d'atteindre des performances en temps réel lorsque la carte recense plus de 15 lieux. D'autres approches nécessitent au contraire une phase d'apprentissage hors-ligne,

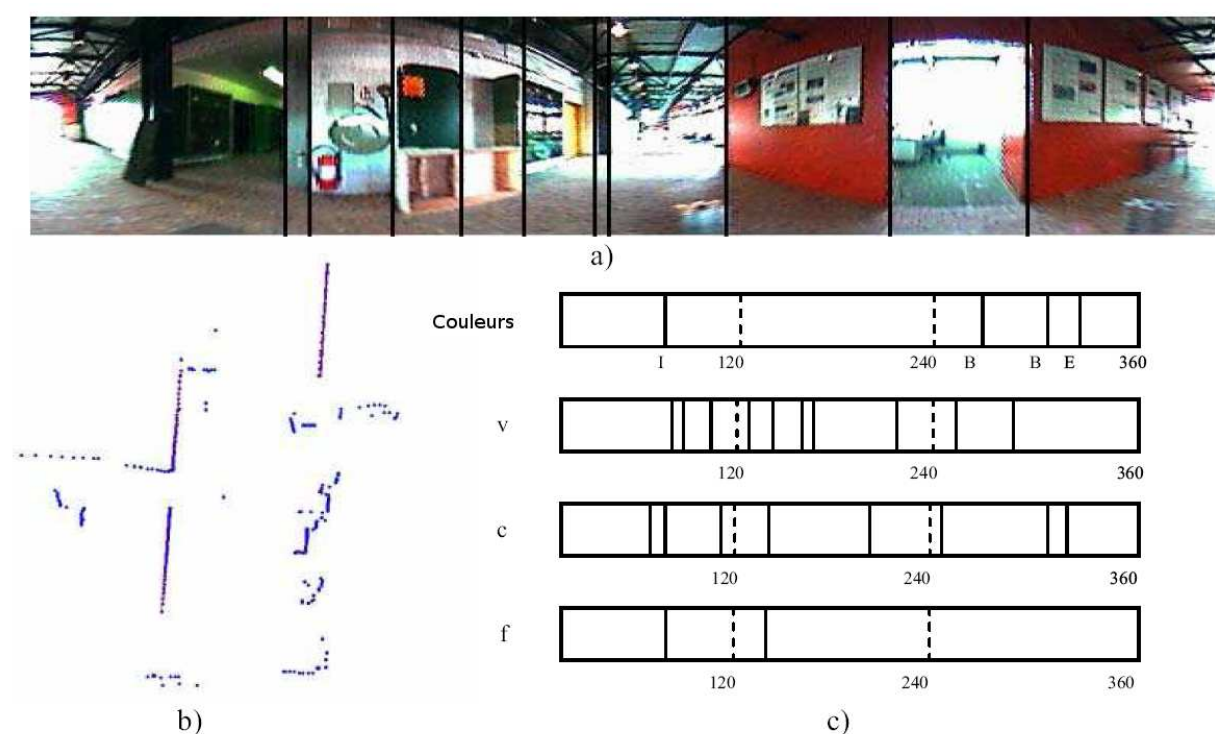


FIG. 5.3: Illustration de la génération d'une empreinte visuelle pour la caractérisation des lieux dans [Tapus and Siegart, 2005]. a) Image provenant d'une caméra omnidirectionnelle dans laquelle on extrait deux types de primitives : des imagerie de couleur et des arêtes verticales. b) Scan laser 360° dans lequel on extrait des coins. c) Génération de l'empreinte visuelle par la combinaison des primitives décrites en a) et b). Chaque bande donne l'angle entre 0° et 360° de ces primitives. Pour les imagerie de couleur (première bande), chaque teinte est identifiée par une lettre entre "A" et "P" (par exemple, "I" pour bleu-clair, "B" pour orange, "E" pour vert-clair). Les arêtes verticales "v" et les coins "c" sont quant à eux respectivement localisés de la même manière dans les deuxième et troisième bandes. La quatrième bande correspond aux co-occurrences pour une orientation donnée d'un coin dans le scan laser et d'une arête verticale dans l'image, identifiables par la lettre "f". En introduisant le terme "n" pour désigner l'absence de primitives sur une longue suite consécutive d'orientations, on obtient ici l'empreinte suivante : clfvnvcvfnvncvnnncvBnvBccE. Source : [Tapus and Siegart, 2005].

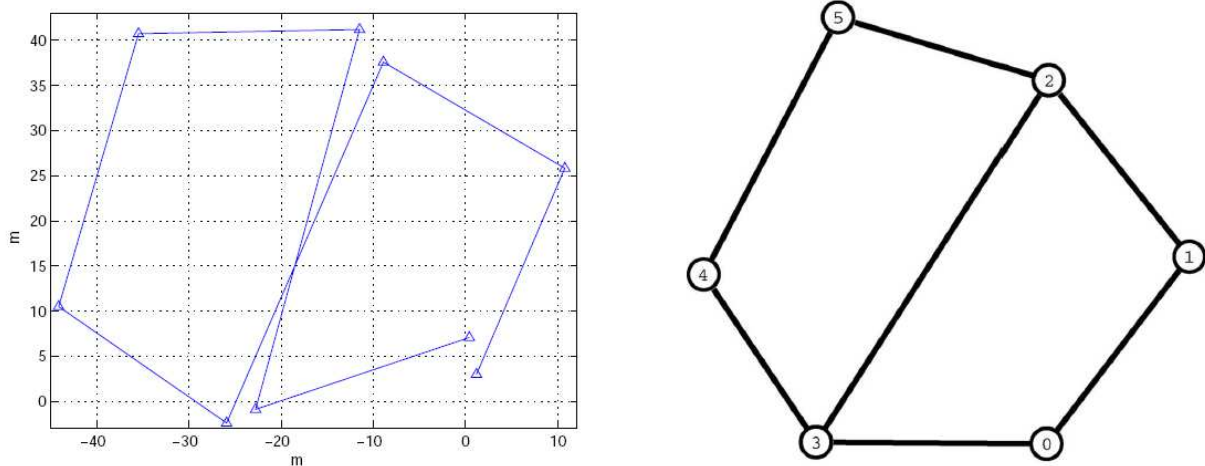


FIG. 5.4: Exemple de carte topologique générée selon la méthode proposée dans [Ranganathan et al., 2006]. Lors du déplacement du robot, six amers distincts conduisent à un total de neuf observations (les positions auxquelles ces observations ont été faites sont localisées dans la partie gauche de la figure grâce à l'odométrie). Le processus de génération de carte incluant l'apparence des amers permet de conclure quels sont les amers qui ont été observé plusieurs fois, aboutissant finalement à la topologie correcte (partie droite de la figure).

ou bien alors contraignent leur application à des environnements répondant à des critères d'apparence, de géométrie et de topologie définis à l'avance.

Afin de mieux cerner les caractéristiques générales des différentes approches recensées dans cet état de l'art, le tableau 5.1 propose un récapitulatif en fonction des critères souhaités, à savoir : l'aspect incrémentiel de l'implémentation, la complexité des traitements, la gestion des fermetures de boucles et la robustesse à l'aliasing perceptuel. Pour chaque critère, une notation (— / — / + / + +) indique à quel point celui-ci est respecté ou s'il peut être atteint facilement. Pour faciliter la compréhension, seules les approches les plus pertinentes ([Cummins and Newman, 2008b], [Fraundorfer et al., 2007], [Goedemé et al., 2007], [Valgren et al., 2007]) sont sélectionnées, ainsi que les familles de méthodes (i.e., POMDP et "Inférence Bayésienne") décrites dans la section 5.3 de ce chapitre.

TAB. 5.1: Récapitulatif.

Méthode	Incrémentiel	Complexité	Gestion FB	Robustesse AP
[Goedemé et al., 2007]	—	+	+	+
[Valgren et al., 2007]	+	—	+	+
[Fraundorfer et al., 2007]	—	++	+	+
[Cummins and Newman, 2008b]	—	+	++	++
POMDP	—	+	—	—
Inférence Bayésienne	++	—	+	+

Pour conclure cet état de l'art et faire le lien avec le tableau donné ci-dessus, nous rappelons les objectifs principaux des travaux rapportés ici. Premièrement, notre méthode est basée sur l'apparence seulement et elle repose sur une caméra monoculaire, alors que la plupart des approches basées sur l'apparence traitent des images omnidirectionnelles ou panoramiques en entrée de leur processus d'estimation. Deuxièmement, le formalisme adopté permet des traitements réalisables de manière complètement incrémentielle et dont la complexité autorise des performances en temps réel. Enfin, nous attachons une importance particulière à la robustesse face à l'aliasing perceptuel, afin de générer des cartes topologiques cohérentes, et ce même en présence de fermetures de boucles dans la trajectoire du robot.

# Chapitre 6

## SLAM topologique

Nous allons dans ce chapitre donner la description du modèle développé dans le cadre de l'application au SLAM topologique. Celui-ci dérive directement de la solution présentée dans la première partie concernant le problème de la détection de fermeture de boucle : c'est pour cela que nous avons choisi de reposer sur un modèle topologique pour la représentation de l'environnement. En effet, pour la détection de fermeture de boucle, l'image courante est comparée à un ensemble de lieux dans le but d'en déterminer la provenance symbolique, et non afin d'inférer une position métrique précise. Comme décrit dans le chapitre 2 de la partie I, le modèle de l'environnement sous-jacent est construit sur la base de l'apparence uniquement, et il ne contient aucune information métrique de localisation. D'autre part, même si les images et les lieux sont caractérisés par un ensemble de primitives locales, selon le paradigme des sacs de mot visuels, la position de ces primitives n'est pas inférée : les mots visuels servent à décrire l'apparence, ignorant toute structure ou géométrie.

Dans l'introduction générale de ce mémoire, nous avons rappelé que la détection de fermeture de boucle était en fait une instance particulière d'un problème plus général : l'association de données. Dans la problématique du SLAM, l'association de données est une partie de la procédure qui permet d'inférer la position actuelle du robot dans la carte. Pour cela, on compare les mesures des capteurs avec l'information contenue dans la carte aux alentours des différentes hypothèses de position maintenues par le processus d'estimation. Suite à la procédure d'association de données, une fois que l'estimation de la position courante est achevée, la carte est mise à jour avec les informations provenant de la vue actuelle. Ainsi, il est possible de reposer sur la détection de fermeture de boucle pour réaliser l'association de données dans le cadre du SLAM topologique. L'environnement est dans ce cas représenté par un graphe dont les noeuds identifient des lieux distincts alors que les arêtes modélisent les relations d'adjacence entre ces lieux. Lorsqu'une nouvelle image est acquise, la détection de fermeture de boucle permet d'en déterminer le lieu de provenance de manière symbolique. Le résultat de cette étape peut être de deux types : soit l'image courante provient d'un lieu connu, soit elle caractérise un lieu encore jamais exploré. Dans le premier cas, la description du lieu auquel

elle appartient doit être mise à jour avec sa caractérisation. Dans le second cas, cette même caractérisation servira de base à la création et à l'identification du nouveau lieu.

## 6.1 Structure de la carte

Ainsi que nous venons de la décrire, la procédure d'association de données considérée dans le cadre du SLAM topologique correspond précisément à la détection de fermeture de boucle abordée comme une tâche de classification d'image. De ce fait, dans l'algorithme mis en oeuvre ici, l'étape de localisation de la caméra consistera à rechercher une fermeture de boucle entre l'image courante et l'ensemble des lieux enregistrés dans le modèle jusque-là. En cas de succès, un lieu sera mis à jour alors que dans le cas contraire, un nouveau lieu sera ajouté, comme décrit dans le chapitre 2 de la partie I. Le processus global conduisant à l'inférence d'une carte cohérente de l'environnement est schématisé par le diagramme de la figure 6.1. Il s'agit de l'adaptation du diagramme présenté dans le cadre de la détection de fermeture de boucle au cas du SLAM topologique. En définitive, l'approche retenue revient à considérer les lieux du modèle  $M_t = \{L_i\}_{i=0}^m$  de l'environnement à l'instant  $t$  de la méthode de détection de fermeture de boucle comme les noeuds de la carte.

### 6.1.1 Information encodée dans les noeuds

Dans la représentation de l'environnement choisie pour la détection de fermeture de boucle (cf. section 2.1 du chapitre 2 de la partie I), un lieu est défini par construction à partir des images qui le décrivent. Ainsi, un noeud de la carte topologique ne représente pas uniquement une image, mais un ensemble d'images provenant de points de vue proches. Cela permet notamment d'obtenir une discrétisation plus efficace de l'environnement, mais également d'encoder la caractérisation de chaque lieu de manière plus détaillée : décrire un lieu par une seule image réduit l'information à une seule vue de ce lieu, ce qui s'avère très limité avec une caméra monoculaire. Ce genre d'approche semble en revanche plus pertinent lorsqu'une caméra panoramique ou omnidirectionnelle est employée, en raison de la meilleure couverture de l'environnement offerte par les images obtenues à partir de ce genre de capteur.

Cependant, en dépit des avantages de la caractérisation des noeuds proposée ici, l'utilisation d'une caméra monoculaire associe implicitement une orientation aux lieux parcourus. En effet, comme cela est constaté dans les résultats expérimentaux (cf. chapitre 7) de cette partie, il semble impossible de reconnaître un lieu lorsque les orientations des différents passages sont trop écartées (i.e., avec une variation de  $90^\circ$  par exemple). Ainsi, plusieurs passages au même endroit avec une variation trop importante de l'orientation du point de vue conduisent à la duplication des noeuds dans la carte : un même lieu est représenté plusieurs fois. Par conséquent, chaque passage dans une zone de l'environnement est conditionné par l'orientation de la caméra lors de ce passage, chaque orientation suffisamment différente des précédentes conduisant à

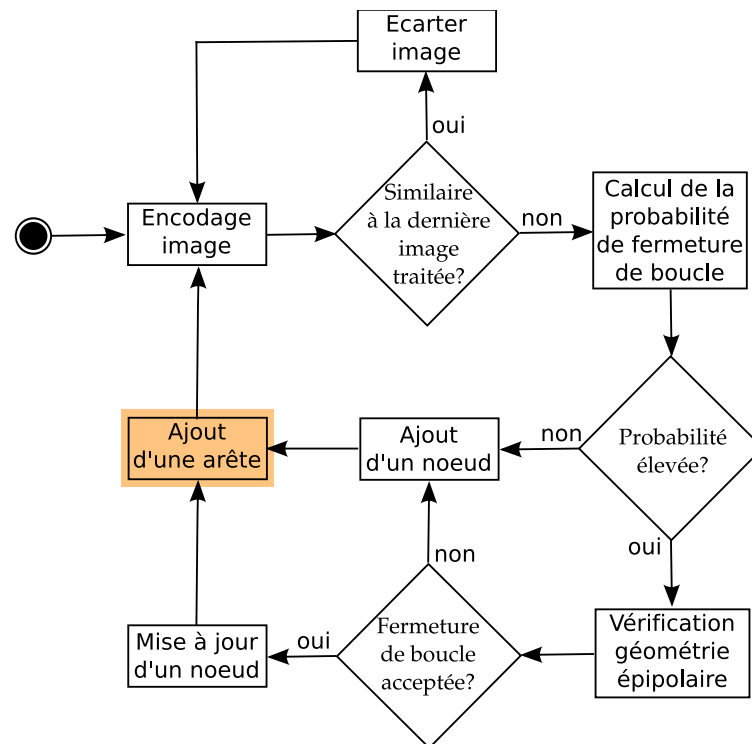


FIG. 6.1: Diagramme du processus global de l'algorithme de SLAM topologique. Il s'agit du même diagramme que pour le processus de détection de fermeture de boucle (voir figure 2.1, chapitre 2 de la partie I), mis à part qu'après chaque ajout ou mise à jour de nœud dans le modèle, une nouvelle arête est créée (cadre sur fond orange).

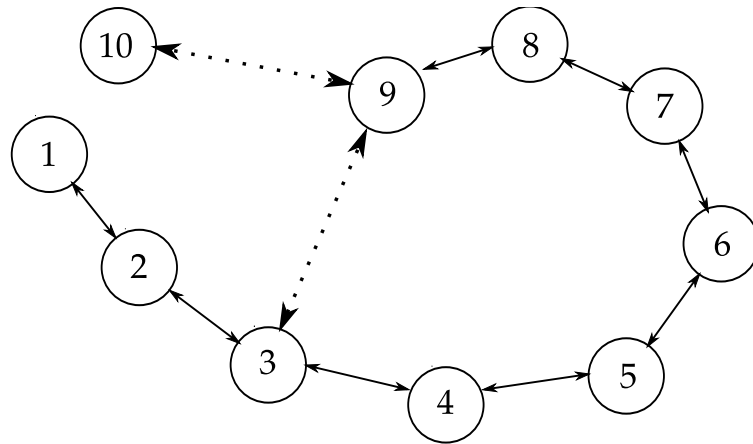


FIG. 6.2: Relation d'adjacence temporelle pour les arêtes de la carte : dans le graphe illustré ici, le nœud 9 est le dernier nœud ajouté. Lorsqu'une nouvelle image sera prise en compte, soit un nouveau nœud sera ajouté (nœud 10), soit un nœud existant sera mis à jour (nœud 3). Dans tous les cas, une nouvelle arête faisant le lien avec le nœud 9 sera ajoutée.

la création d'un nouveau lieu. Toutefois, comme nous le détaillons dans la discussion de cette partie (cf. chapitre 8), l'ajout d'une information métrique dans la carte permettrait de lever ce genre d'ambiguïté.

### 6.1.2 Information encodée dans les arêtes

Dans l'état de l'art de cette partie (cf. section 5), nous avons vu qu'il pouvait y avoir différentes significations pour les arêtes de la carte. D'une manière générale, celles-ci servent à encoder une relation de proximité. Toutefois, il ne s'agit pas forcément de proximité dans l'espace des positions, et il arrive fréquemment que cette notion soit exprimée dans l'espace de l'apparence : deux lieux reliés par une arête seront dans ce cas similaires. Le choix de l'information enregistrée sur les arêtes est en général déterminé en fonction de l'utilité finale de la carte : la navigation. Ainsi, les arêtes indiquent les lieux semblables au lieu courant et accessibles dans un voisinage proche, ou bien simplement les lieux qui ont été traversés avant et après le nœud courant lors d'un passage antérieur à cet endroit. Dans ce dernier cas, on enregistre uniquement une information binaire d'adjacence temporelle entre les nœuds : chaque arête indique que les lieux qu'elle relie ont été parcourus de manière consécutive dans le temps. Pour la navigation, cette information permet de savoir quels nœuds sont accessibles depuis la position courante, sans pour autant enregistrer de route à suivre pour atteindre ces lieux.

Dans le cadre des travaux rapportés ici, c'est ce type d'information d'adjacence temporelle qui sera retenue pour définir les arêtes de la carte : celles-ci encodent alors l'ordre temporel dans lequel les lieux sont traversés par la caméra. En conséquence, lorsqu'une image est traitée, le nœud qu'elle permettra de mettre à jour ou de créer sera lié au dernier nœud ajouté ou mis à jour dans la carte (voir figure 6.2).



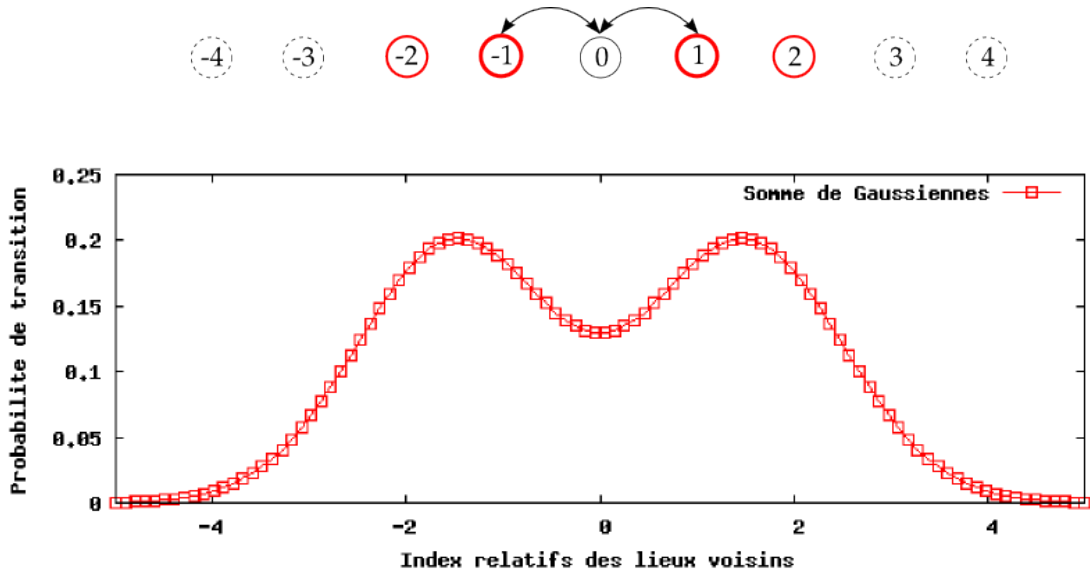


FIG. 6.3: Relations entre noeuds d'après le modèle de transition et selon les arêtes de la carte. Le modèle de transition lie par un coefficient non négligeable un noeud à 4 de ses voisins dans le temps (l'épaisseur du cercle rouge entourant chaque noeud dépend de la probabilité de transition, les noeuds entourés de pointillés recevant une probabilité négligeable). Le choix retenu ici pour les arêtes associe un noeud à ces plus proches voisins dans le temps (le noeud 0 n'est relié avec un trait noir qu'aux noeuds -1 et 1).

### Arêtes et voisinage

Une nouvelle relation de voisinage apparaît donc sous la forme des arêtes, en plus des relations déjà établies entre les lieux pour les besoins de la détection de fermeture de boucle. Plus précisément, ces relations sont encodées par la somme de Gaussienne exprimant la probabilité de transition d'un lieu à l'autre. Ce modèle de transition, défini dans la section 2.3.1 du chapitre 2 (partie I), permet de prédire le lieu de l'image acquise à l'instant  $t$  compte tenu de la probabilité de localisation à l'instant  $t - 1$ . La définition de ce modèle est basée sur l'existence de similarités entre les images acquises consécutivement dans le temps, induisant une relation de voisinage temporel entre les lieux du modèle de l'environnement. Dans le cadre de l'application au SLAM topologique, le modèle de l'environnement est une carte, et les lieux  $y$  sont définis comme des noeuds. Ainsi, cette relation de voisinage lie les noeuds de la carte par les similarités qu'ils partagent. Cependant, d'après ce modèle de transition, un noeud est lié à au moins quatre voisins avec un coefficient non négligeable, alors que les arêtes telles que définies ici n'associent un noeud qu'à deux de ces voisins (voir figure 6.3).

D'après la figure 6.3, on remarque qu'il existe une cohérence entre les arêtes (encodant l'ordre d'enchaînement des noeuds à des fins de navigation), et les probabilités de transition (définies sur la base des similarités existant entre images consécutives) : plus on s'écarte du lieu courant, plus la probabilité de transi-

tion est faible, et plus le nombre d'arêtes à traverser augmente. Dit autrement, pour joindre un noeud éloigné du noeud courant, il faudra traverser plusieurs noeuds intermédiaires, et plus on s'écartera du noeud courant, plus la similarité avec celui-ci diminuera.

Cependant, pour la navigation, il est important de ne pas considérer ici toutes les relations induites par le modèle de transition et provenant des similarités entre images comme des arêtes dans la carte. En effet, nous avons mentionné dans l'état de l'art de cette partie des travaux ([Booi et al., 2007]) où l'édition de liens entre les noeuds dans la carte était déterminée sur la base d'une mesure de similarité précise, nécessitant la comparaison explicite des images correspondantes. Dans notre cas, cette édition de lien est réalisée sur la base de l'adjacence temporelle, et une mesure exacte du coefficient de similarité entre les images correspondantes n'est pas effectuée. De plus, dans les applications définissant les arêtes de cette manière, les auteurs emploient généralement une caméra omnidirectionnelle, ce qui permet d'obtenir un bon taux de recouvrement entre les différentes images, même si celle-ci sont acquises de points de vue distants : grâce à ce recouvrement, des lieux relativement éloignés partageront quelques similarités, justifiant de fait le choix d'une telle information pour la définition des arêtes. Dans notre cas, en choisissant d'ajouter une arête dans la carte chaque fois que deux noeuds sont liés par une probabilité de transition non négligeable, nous introduirions une route plausible entre les lieux correspondants, sans avoir l'assurance qu'une telle jonction directe est effectivement possible. C'est pourquoi nous limitons ces liens aux voisins directs uniquement.

## 6.2 Disposition du graphe

Dans l'application exposée ici, l'environnement est représenté sous la forme d'un graphe liant des noeuds correspondants aux différents lieux de l'environnement. Grâce à cette modélisation sous la forme d'un graphe, et en raison de la présence d'arêtes joignant les noeuds, il est possible d'adapter la disposition du graphe afin de rendre compte explicitement de la nature cyclique de la trajectoire du robot dans l'environnement lors d'une fermeture de boucle. Pour cela, on considère le graphe comme un réseau de ressorts (i.e., les arêtes) dont les points de jonction (i.e., les noeuds) sont localisés dans l'espace. Pour une meilleure lisibilité du graphe, on ne considère ici que des positions dans le plan 2D. Ainsi, il est possible de définir une notion d'équilibre dans ce réseau de ressorts, qui revient à rechercher les positions des points de jonction qui permettent de satisfaire au mieux les contraintes exercées par les ressorts. On définit alors pour chaque ressort une longueur à vide qui correspond à sa longueur lorsqu'il est libre (i.e., non contraint). La position d'équilibre recherchée correspond en fait au minimum d'énergie potentielle du réseau de ressorts, et donc à la position des points de jonction pour laquelle l'écart de la longueur des ressorts par rapport à leur longueur à vide est la plus faible. La méthode de "relaxation" employée ici pour trouver la position d'équilibre du réseau de ressorts est décrite dans les travaux de [Kamada and Kawai, 1989]. Une illustration du principe de fonctionnement de cette méthode pour l'application à la construction de carte topologique est donnée dans la figure 6.4.

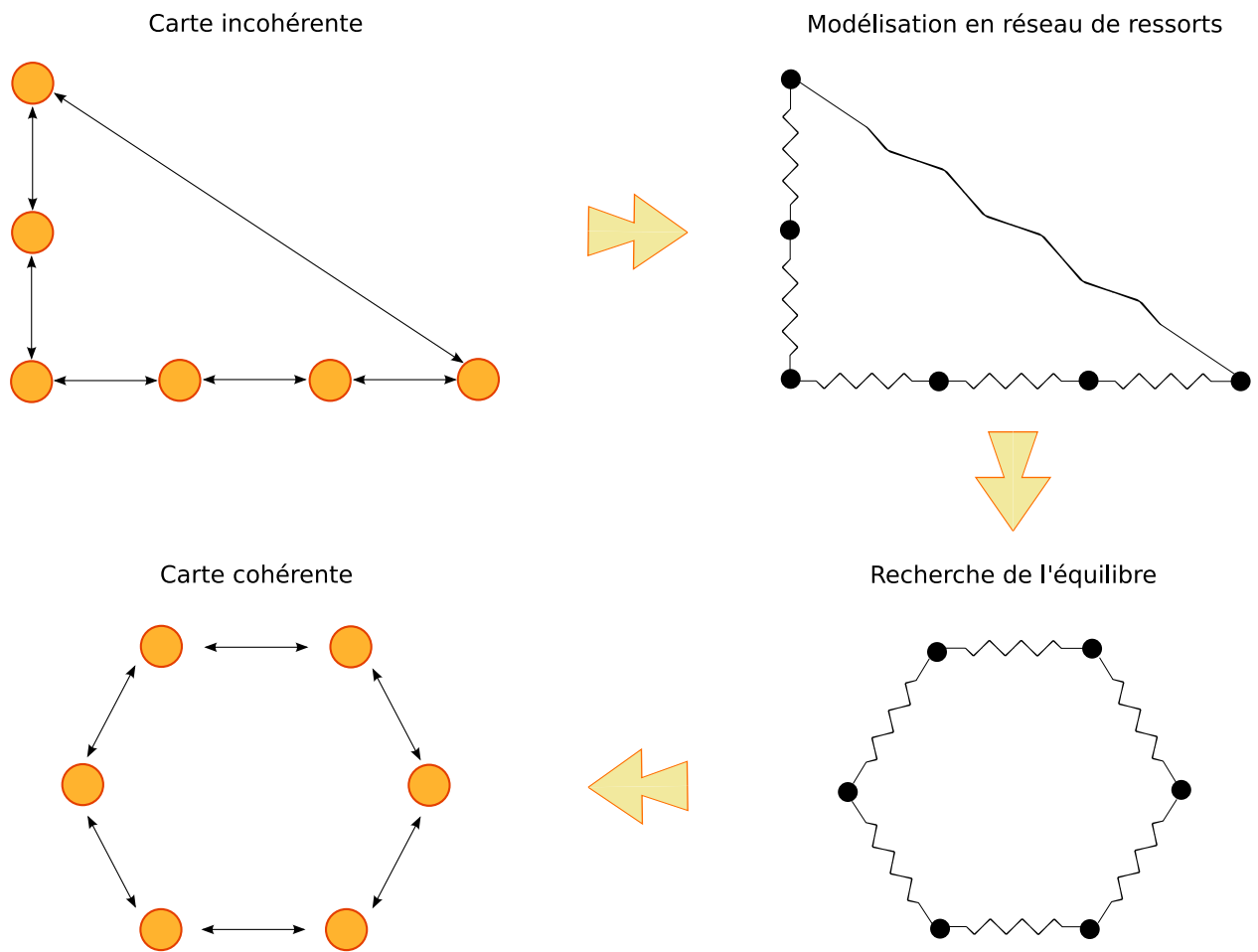


FIG. 6.4: Recherche de la position d'équilibre dans le réseau de ressorts pour obtenir une carte cohérente en cas de fermeture de boucle. Lors d'une fermeture de boucle, si la relaxation n'est pas appliquée au réseau de ressorts, la carte est incohérente. En recherchant la position d'équilibre du réseau visant à minimiser l'écart entre les longueurs des ressorts et leurs longueurs à vide, on obtient une carte cohérente.

Dans notre cas, la longueur à vide pour les ressorts est définie par l'information contenue dans les arêtes de la carte : l'existence d'une arête liant deux noeuds correspond à la présence d'un ressort de longueur à vide unitaire entre les points de jonctions que représentent ces noeuds. Il est important de noter ici que la méthode de relaxation présentée ci-dessus n'est utile que lorsque le graphe est connecté (i.e., il faut qu'il présente au moins un cycle dans sa structure). Si le graphe n'est pas connecté, la relaxation disposera les noeuds en ligne droite, puisque les arêtes ne contiennent pas d'information d'orientations relatives.

### 6.3 Ajout d'une information métrique

Les travaux présentés dans cette partie concernent la construction de cartes topologiques sur la base de l'apparence uniquement. Notre solution repose pour cela sur la méthode de détection de fermeture de boucle décrite dans la première partie de ce mémoire. Toutefois, comme nous l'avons mentionné dans l'état de l'art de cette partie (cf. section 5.1, page 96), il est possible d'ajouter une information métrique dans une carte topologique. Cela permet alors d'estimer une position précise pour chaque noeud, sous la forme de coordonnées cartésiennes. D'une manière générale, l'ajout d'une information métrique dans la carte offre l'avantage majeur de pouvoir mémoriser des déplacements précis entre les différents noeuds, facilitant par conséquent la navigation entre ces noeuds. Dans une perspective expérimentale, nous avons adapté simplement notre modèle de SLAM topologique pour intégrer ce type d'information dans la carte, reposant pour cela sur une plateforme mobile fournissant une mesure de l'odométrie correspondant aux déplacements du robot. Ces travaux, au caractère préliminaire, ont été réalisés en collaboration avec Nicolas BEAUFORT dans le cadre d'un stage réalisé au sein du laboratoire [Beaufort].

#### 6.3.1 Relaxation pour l'estimation de la position des noeuds

L'information utilisée dans notre cas provient des mesures d'odométrie fournies par le robot. Ces mesures donnent une estimation du déplacement relatif du robot entre deux positions  $i$  et  $j$ , sous la forme de l'angle relatif  $\phi_{ij}$  les séparant et d'un vecteur de déplacement (caractérisé par sa longueur  $d_{ij}$  et son angle  $\theta_{ij}$  dans le repère de la position  $i$ ). Ce sont ces informations qui seront encodées dans les arêtes de la carte et utilisées pour estimer la position des noeuds (voir figure 6.5).

On enregistre donc sur les arêtes une information de positionnement relatif des noeuds. A partir de cette information, il est possible d'obtenir une estimation de la position métrique de chaque noeud  $i$ , sous la forme de coordonnées cartésiennes 2D plus une orientation  $(x_i, y_i, \theta_i)$  exprimées dans un référentiel absolu. Pour cela, nous employons une méthode simple de relaxation proposée par les auteurs de [Duckett et al., 2000]. Celle-ci a été développée spécifiquement pour la relaxation à partir d'informations de distance et d'orientation : c'est pourquoi nous l'avons préféré à la méthode plus basique utilisée pour la disposition du graphe (cf. section 6.2) qui ne prend en compte qu'une notion de poids sur les arêtes. La méthode retenue ici avait par ailleurs déjà été réutilisée dans le même contexte qu'ici dans [Filliat, 2001]. L'idée générale est

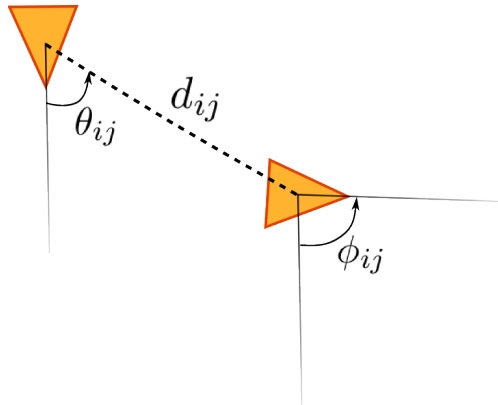


FIG. 6.5: Ajout d'une information métrique d'odométrie sur les arêtes de la carte. Entre les positions  $i$  et  $j$ , l'odométrie mesure un déplacement relatif donné par l'angle  $\phi_{ij}$  et par le vecteur de déplacement 2D caractérisé par  $d_{ij}$  et  $\theta_{ij}$ . Cette information est alors ajoutée sur l'arête de la carte liant les noeuds  $i$  et  $j$ .

de parcourir, pour chaque noeud du graphe, l'ensemble de ses voisins (i.e., l'ensemble des noeuds auxquels il est directement lié). On calcule alors, à partir de chacun des voisins, une position absolue pour le noeud considéré, en composant simplement la position du voisin avec l'information d'odométrie le liant au noeud considéré. On obtient donc, pour chaque noeud, un ensemble de positions calculées à partir des positions des noeuds voisins. On définit finalement la position du noeud considéré comme étant la moyenne des positions ainsi calculées. Cet algorithme, dont le pseudo-code est donné dans le tableau 6.3.1, est itéré jusqu'à ce que les variations des positions des noeuds soient négligeables.

---

#### Relaxation

---

```

tantque le graphe  $G$  n'a pas convergé faire
  pour tous les noeuds  $N_i \in G$  sauf le premier faire
    pour tous les voisins  $V_j$  de  $N_i$  faire
       $\text{Position}_j(N_i) = \text{Position}(V_j) + \Delta_{ji}$ 
    fin pour
     $\text{Position}(N_i) = \frac{\sum_j \text{Position}_j(N_i)}{\text{nb\_voisins}(N_i)}$ 
  fin pour
fin tantque

```

---

TAB. 6.1: Pseudo-code pour l'algorithme de relaxation.  $\Delta_{ji}$  désigne le déplacement relatif permettant de passer du noeud  $V_j$  au noeud  $N_i$  (i.e., il s'agit de l'information fournie par l'odométrie, voir figure 6.5.)

L'utilisation d'un algorithme de relaxation pour l'estimation de la position des noeuds permet de corriger la dérive de l'odométrie. En effet, l'intégration des mesures d'odométrie a des conséquences dramatiques

sur l'estimation de la position des noeuds de la carte : en intégrant ces mesures, on intègre également les erreurs sur ces mesures. Par conséquent, la position estimée pour les noeuds diverge rapidement de leur valeur réelle. En reposant sur un algorithme de relaxation pour l'inférence de la position des noeuds, on fusionne les estimations obtenues à partir de plusieurs noeuds pour calculer la position d'un noeud donné : on filtre ainsi le bruit sur les mesures d'odométrie, et la position inférée est plus cohérente à long terme.

Enfin, en cas de fermeture de boucle, l'information d'adjacence apportée par la fermeture du cycle permet d'améliorer la cohérence globale des positions estimées pour les noeuds de la carte grâce à l'algorithme de relaxation. En effet, la fermeture de boucle va créer un lien entre le noeud de localisation du robot à l'instant  $t - 1$  et le noeud qui vient d'être reconnu. Il est donc possible d'inférer une topologie et une estimation de la position des noeuds qui soient cohérentes à la fois avec l'apparence des lieux (i.e., par la reconnaissance d'un lieu passé), et avec les positions relatives de ces lieux (i.e., grâce à l'information d'odométrie encodée sur les arêtes correspondantes). L'amélioration dans l'estimation des positions des noeuds est illustrée dans la figure 6.6.

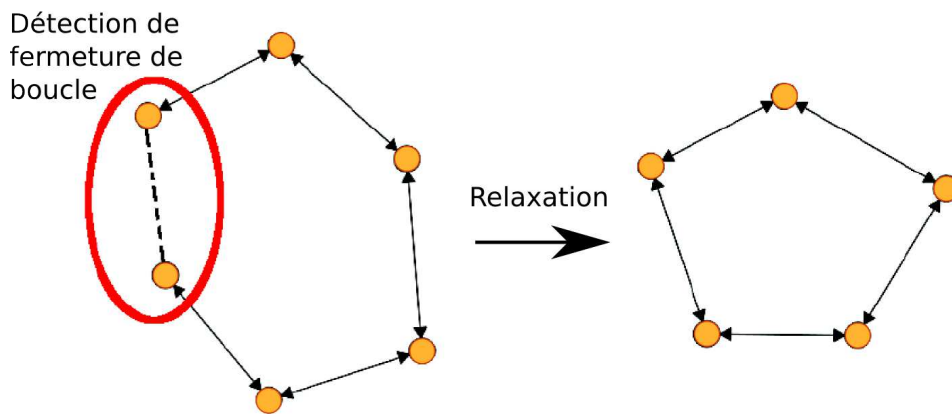


FIG. 6.6: Illustration de l'amélioration de l'estimation de position des noeuds par l'algorithme de relaxation lorsqu'une fermeture de boucle est détectée. En détectant une fermeture de boucle entre les noeuds présents dans l'ellipse rouge (i.e., ces noeuds correspondent au même lieu), on introduit une nouvelle contrainte dans la carte qui permet d'améliorer la topologie estimée par l'algorithme de relaxation.

# Chapitre 7

## Résultats expérimentaux

Ce chapitre est dédié aux résultats expérimentaux obtenus dans le cadre de l'application au SLAM topologique. Notamment, nous démontrons la qualité de notre approche par la construction de cartes topologiques en-ligne et en temps réel dans divers environnements sur la base d'une simple caméra monoculaire uniquement. Ces résultats ont été obtenus à partir de deux séquences d'images : la première correspond à une acquisition réalisée en intérieur uniquement, alors que la seconde contient à la fois des images d'intérieur et d'extérieur. Dans les deux cas, les environnements choisis présentent un aliasing perceptuel important, afin de prouver la robustesse de notre solution. Avec la première séquence, nous montrons qu'il est possible d'obtenir des résultats satisfaisants avec une seule caractérisation pour les images (i.e., grâce aux primitives SIFT), à condition que celles-ci présentent assez de texture. Cela permet notamment de limiter les traitements afin d'augmenter la rapidité générale. Pour la séquence "mixte" (i.e., intérieur / extérieur) en revanche, les deux espaces de représentation sont nécessaires afin d'obtenir une carte cohérente de l'environnement. Chacune des deux séquences est présentée séparément, dans une section dédiée. Par ailleurs, une section supplémentaire donne des résultats préliminaires en ce qui concerne l'ajout d'information métrique dans la carte. Les résultats exposés dans ce chapitre ont notamment fait l'objet d'une publication ([Angeli et al., 2008c]).

### 7.1 Environnement d'intérieur

Dans cette section, nous présentons des résultats obtenus avec un seul espace de représentation (i.e., les primitives SIFT). Dans l'environnement considéré, cela s'avère particulièrement efficace car les traitements sont réduits du fait de la caractérisation unique des images, et parce que la carte obtenue est cohérente avec la trajectoire suivie par la caméra. Les images considérées ici ont été extraites d'une vidéo durant 247 secondes avec une fréquence d'une image par seconde. Pour l'acquisition, nous avons utilisé une simple caméra monoculaire grand-angle (angle de vue de  $120^\circ$ ), avec gestion automatique de l'exposition. Un

seul espace de représentation (i.e., les primitives SIFT) est pris en compte pour caractériser les images de 320x240 pixels. La figure 7.5 donne un exemple des images composant la séquence. Encore une fois, la caméra était tenue à la main lors des déplacements, et plusieurs cycles ont été réalisés dans sa trajectoire.



FIG. 7.1: Exemples d'images provenant de la séquence d'intérieur : les images sont ordonnées par ordre d'acquisition, suivant le déplacement de la caméra lors de l'expérience.

Afin de démontrer la qualité de notre solution pour le SLAM topologique, les résultats sont présentés sous la forme de la carte obtenue à la fin de l'expérience. Celle-ci est donnée dans la figure 7.2, à côté du plan du bâtiment dans lequel l'acquisition a été effectuée. Pour pouvoir mettre aisément en correspondance les noeuds de la carte avec le plan, un code couleur est mis en place afin d'identifier les différentes zones géographiques de passage de la caméra, comme précédemment.

La partie gauche de la figure 7.2 donne une illustration de la trajectoire de la caméra superposée sur le plan du bâtiment dans lequel a été conduite l'expérience. Le parcours débute par une première boucle dans la zone bleue, avant d'entrer dans la zone magenta. Après cela, la caméra retourne dans la zone bleue et se dirige tout droit vers la zone rouge. Elle revient alors encore une fois dans la zone bleue avant de découvrir la zone verte. Ce trajet se termine à côté du 8<sup>ème</sup> cercle blanc.

Dans la partie droite de la figure 7.2, la carte topologique correspondant au trajet de la caméra décrit ci-dessus est donnée. Afin d'en faciliter la compréhension, le même code couleur y est employé, permettant



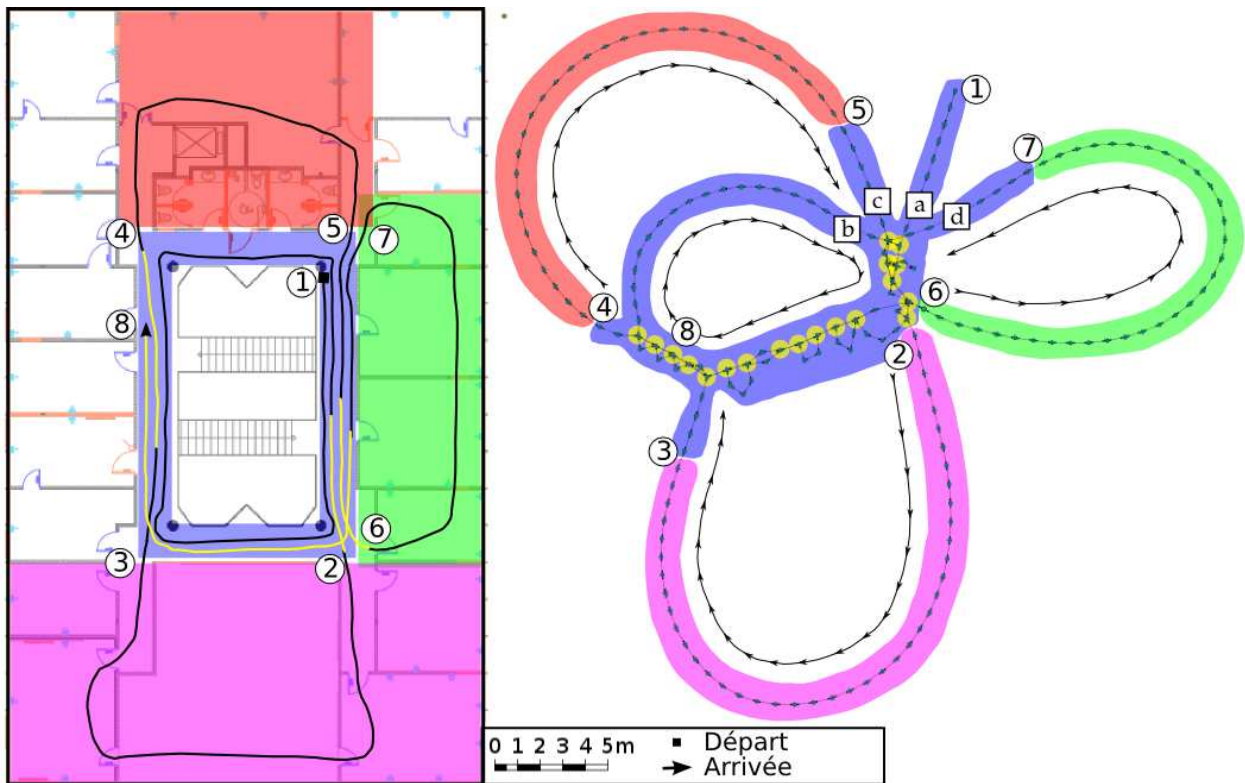


FIG. 7.2: Plan du bâtiment de l'expérience avec la trajectoire de la caméra en superposition (partie gauche de la figure) et carte topologique associée (partie droite de la figure). La disposition du graphe est obtenue grâce à un simple algorithme de relaxation [Kamada and Kawai, 1989]. Les détails concernant la trajectoire de la caméra et les détections de fermeture de boucle correspondantes sont donnés dans le texte.

ainsi d'identifier facilement les différentes zones cartographiées. On peut rapidement s'apercevoir que tous les cycles correspondant à un retour de la caméra dans la zone bleue sont correctement détectés, de même que ceux réalisés à l'intérieur de cette zone. Les événements de détection de fermeture de boucle peuvent être identifiés par la couleur jaune de la trajectoire dans le plan, ainsi que par la coloration jaune des noeuds correspondants dans la carte. Il est important de relever ici que plusieurs événements de ce genre peuvent être représentés par un seul même noeud. Par exemple, la caméra est passée quatre fois aux alentours du 6<sup>ème</sup> cercle blanc, provoquant plusieurs détections de fermeture de boucle parmi les noeuds avoisinant : en conséquence, ces noeuds encodent les différents passages de la caméra, et ils sont de ce fait caractérisés par plusieurs images.

Lorsque l'on considère la carte topologique obtenue plus en détails, on peut remarquer qu'il existe un délai entre l'occurrence d'une fermeture de boucle et sa détection. Cela est particulièrement remarquable chaque fois que la caméra retourne dans la zone bleue : le véritable noeud de fermeture de boucle (i.e., le

noeud correspondant au retour effectif de la caméra dans un lieu connu) précède le noeud de fermeture de boucle tel que sélectionné par l’algorithme de SLAM. Par exemple, lorsque la caméra retourne de la zone magenta vers la zone bleue, la transition effective, à côté du 3<sup>ème</sup> cercle blanc, n’est détectée que trois noeuds plus tard, au niveau du noeud entouré de jaune qui lie les zones magenta et bleue dans la carte. La raison de ce délai a déjà été discutée dans le chapitre 3 de la partie I : il s’agit de la faible réactivité du modèle de probabilité. Cette faible réactivité a pour effet néfaste de rendre certaines situations de fermetures de boucles ambiguës. En effet, quand la caméra est déplacée consécutivement dans plusieurs lieux déjà visités (i.e., “d” dans la figure 7.2), les noeuds qui correspondent aux passages antérieurs de la caméra obtiennent une vraisemblance comparable (i.e., “a”, “b” et “c” dans la figure 7.2). En conséquence, ces pics d’amplitude similaire (voir figure 7.3) empêchent la probabilité a posteriori d’être concentrée de façon non-ambiguë sur une hypothèse en particulier. Toutefois, l’acquisition future de quelques images suffit à lever l’ambiguïté (i.e., lorsque la caméra atteint le noeud de fermeture de boucle qui joint les branches “a”, “b”, “c” et “d” dans la figure 7.2).

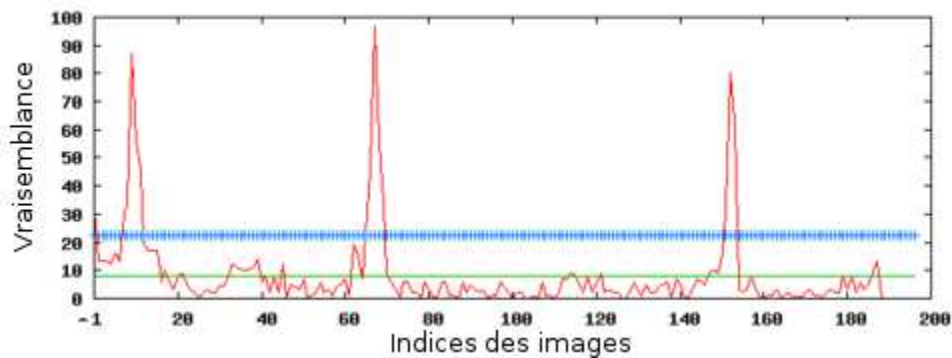


FIG. 7.3: Un exemple de vraisemblance ambiguë : cela correspond à une situation où la caméra retourne pour la quatrième fois (“d” dans la figure 7.2) dans un lieu déjà visité. De gauche à droite, les pics correspondent aux passages antérieurs “a”, “b” et “c” de la caméra dans la figure 7.2. L’acquisition de nouvelles images permettra de lever l’ambiguïté.

### 7.1.1 Performances

Dans les résultats expérimentaux présentés ci-dessus, la totalité des traitements a été réalisée en-ligne et en temps réel : 123s ont été nécessaires pour traiter les 247s de la séquence, avec un Pentium Core2 Duo 2.33GHz et pour des images de taille 320x240 pixels. La figure 7.4 donne l’évolution des temps de calcul par image au cours du temps. On peut notamment remarquer, en comparant cette figure à la figure 7.10, que le temps d’extraction des primitives SIFT dans la séquence intérieur est sensiblement équivalent à ce temps dans la séquence mixte. Ceci est dû au nombre plus important de primitives SIFT extraites dans les images de la séquence d’intérieur (avec la même taille d’image) : c’est grâce à cette multitude de primitives

qu'un seul espace de représentation est suffisant ici, alors que la prise en compte des histogrammes  $H$  est indispensable dans la séquence mixte pour obtenir une carte cohérente.

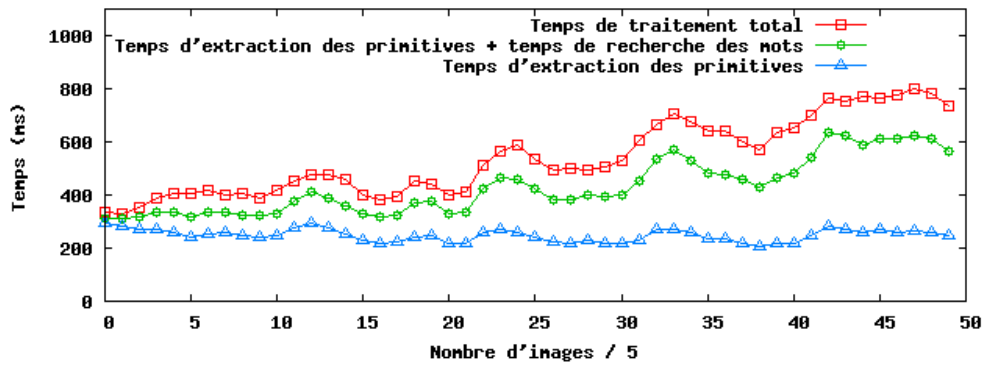


FIG. 7.4: Évolution des temps de traitements par image : la figure donne le temps requis pour extraire les primitives dans les images (triangles), auquel est ajouté le temps nécessaire à la recherche des mots correspondants dans le vocabulaire (cercles), avec enfin le temps total de traitement par image (carrés). Pour améliorer la lisibilité, les temps de calcul ont été moyennés toutes les 5 images.

TAB. 7.1: Performances.

Longueur	#img	CPU	#SIFT	#FB	%VP	#FA
4m07s	247	2m03s	24156	87	59	0

Le tableau 7.1 recense des informations telles que la longueur de la séquence, le nombre d'images qu'elle contient, le temps CPU requis pour la traiter, la taille du dictionnaire SIFT à la fin de l'expérience ("#SIFT", en nombre de mots), le nombre de fermetures de boucles ("#FB", déterminé à la main d'après la trajectoire de la caméra), le taux de vrais positifs ("%VP", les fermetures de boucles correctement détectées), et le nombre de fausses alarmes ("#FA", hypothèses erronées qui reçoivent une probabilité élevée mais qui sont écartées par l'algorithme de géométrie multi-vues). On observe un taux de vrais positifs (i.e., 59%) relativement faible comparé aux expériences précédentes (cf. chapitre 3 de la partie I). Comme cela a été exposé plus haut (voir notamment la figure 7.3), certaines fermetures de boucles tardent à être détectées en raison des ambiguïtés introduites par les différents passages de la caméra en un même lieu. Par conséquent, le taux de vrais positifs s'en retrouve dégradé. D'autre part, en dépit de l'important aliasing perceptuel présent dans l'environnement (cf. figure 7.1), on remarque que le nombre de fausses alarmes est nul, ce qui confirme encore une fois la robustesse de notre solution.



FIG. 7.5: Exemples d'images provenant de la séquence mixte : les images sont ordonnées par ordre d'acquisition, suivant le déplacement de la caméra lors de l'expérience.

## 7.2 Environnement mixte

Dans cette section, les résultats présentés ont été obtenus à partir d'une séquence d'images acquises dans un environnement mixte, mêlant scènes d'intérieur et d'extérieur (cf. figure 7.5 pour un aperçu des images provenant de la séquence). Pour cela, la même caméra que dans l'expérience précédente a été déplacée à la main dans un environnement en suivant une trajectoire au cours de laquelle plusieurs cycles ont été effectués. Ici, les deux espaces de représentation sont pris en compte (i.e., primitives SIFT et histogrammes de teinte) pour caractériser les images de taille 320x240 pixels. La longueur totale de la séquence est de 415 secondes et les images y sont extraites avec une fréquence de 0.5Hz : la profondeur de champ des scènes observées ici offre la possibilité de réduire le nombre d'images par seconde, même pour les passages en intérieur étant donné que ceux-ci correspondent à des couloirs rectilignes.

La figure 7.6 donne un aperçu de la trajectoire de la caméra ainsi que la carte qu'elle a permis de construire. Dans la partie de gauche de la figure, les trois vues aériennes correspondent aux différents niveaux des deux bâtiments (i.e., B31 and B41) autour et à l'intérieur desquels l'acquisition a été réalisée. Au premier niveau, seulement des images d'extérieur ont été enregistrées, dans la zone rouge qui entoure les bâtiments. D'autres images d'extérieur ont été acquises lors de passages entre les bâtiments, en empruntant la passerelle violette du 2<sup>ème</sup> étage, ainsi que lors des changements de niveau, étant donné que les escaliers (identifiés par des carrés bleus) sont situés à l'extérieur des bâtiments. Les images d'intérieur ont quant à elle été obtenues dans le bâtiment B41 exclusivement, aux deuxième (zone vert-foncé) et troisième (zone vert-clair) étages. Des exemples de ces images d'intérieur sont donnés dans la figure 7.7, afin de se rendre compte de leur similarité tant en termes d'apparence que de structure.

La partie de droite de la figure 7.6 correspond à la carte qui a été construite de façon incrémentielle en analysant la séquence d'images décrite ci-dessus (la disposition du graphe a été obtenue grâce à un simple algorithme de relaxation [Kamada and Kawai, 1989], voir section 6.2 du chapitre 6). Dans cette carte, les noeuds de fermeture de boucle sont entourés de jaune. Lors de son parcours, la caméra a effectué plusieurs cycles en intérieur et en extérieur, et ce sur les différents niveaux des bâtiments. Il serait donc difficile d'expliquer étape par étape la trajectoire correspondante. En conséquence, une analyse qualitative de ce parcours est fournie à la place.

Au cours de cette expérience, la plupart des cycles ont été réalisés en extérieur au premier niveau coloré en rouge, en intérieur aux deux niveaux correspondant aux zones vertes, mais également lors de l'emprunt des escaliers "b" et "c" qui joignent ces deux niveaux. Ceci est remarquable dans le graphe par la présence de noeuds jaunes de fermeture de boucle dans les zones rouges, vertes et bleues correspondantes. En particulier, il est important de noter qu'en dépit de l'important niveau d'aliasing perceptuel existant entre les deux niveaux d'intérieur (voir figure 7.7), aucune erreur d'association n'a été faite et ceux-ci ont n'ont pas été confondus. De plus, il y a certaines parties de l'environnement qui ne sont visitées qu'une seule fois, en conséquence de quoi celles-ci sont exemptes de noeuds de fermeture de boucle dans le graphe : c'est notamment le cas pour la plupart des escaliers, "b" et "c" mis à part, ainsi que pour la longue courbe dans la partie haute du graphe qui correspond à un passage unique au Nord du bâtiment B31.

Par ailleurs, les deux niveaux verts d'intérieur sont complètement traversés plusieurs fois. Cependant, en prenant garde aux parties correspondantes dans le graphe, on remarque qu'une séparation apparaît dans la zone vert-clair du troisième étage (i.e., au niveau du cercle blanc "1") : une partie de ce niveau n'a pas été reconnue correctement et a donc été enregistrée deux fois dans la carte. Ceci est dû à la présence d'une personne devant la caméra alors que celle-ci était déplacée dans un couloir étroit : l'obstruction du champs de vision était trop importante pour permettre la validation de la contrainte de géométrie épipolaire (i.e., en raison de l'obstruction, l'association de primitives locales nécessaire à l'algorithme de géométrie multi-vues n'a pu être faite correctement). En conséquence, la détection de fermeture de boucle n'a pas réussi et certains lieux ont été dupliqués.



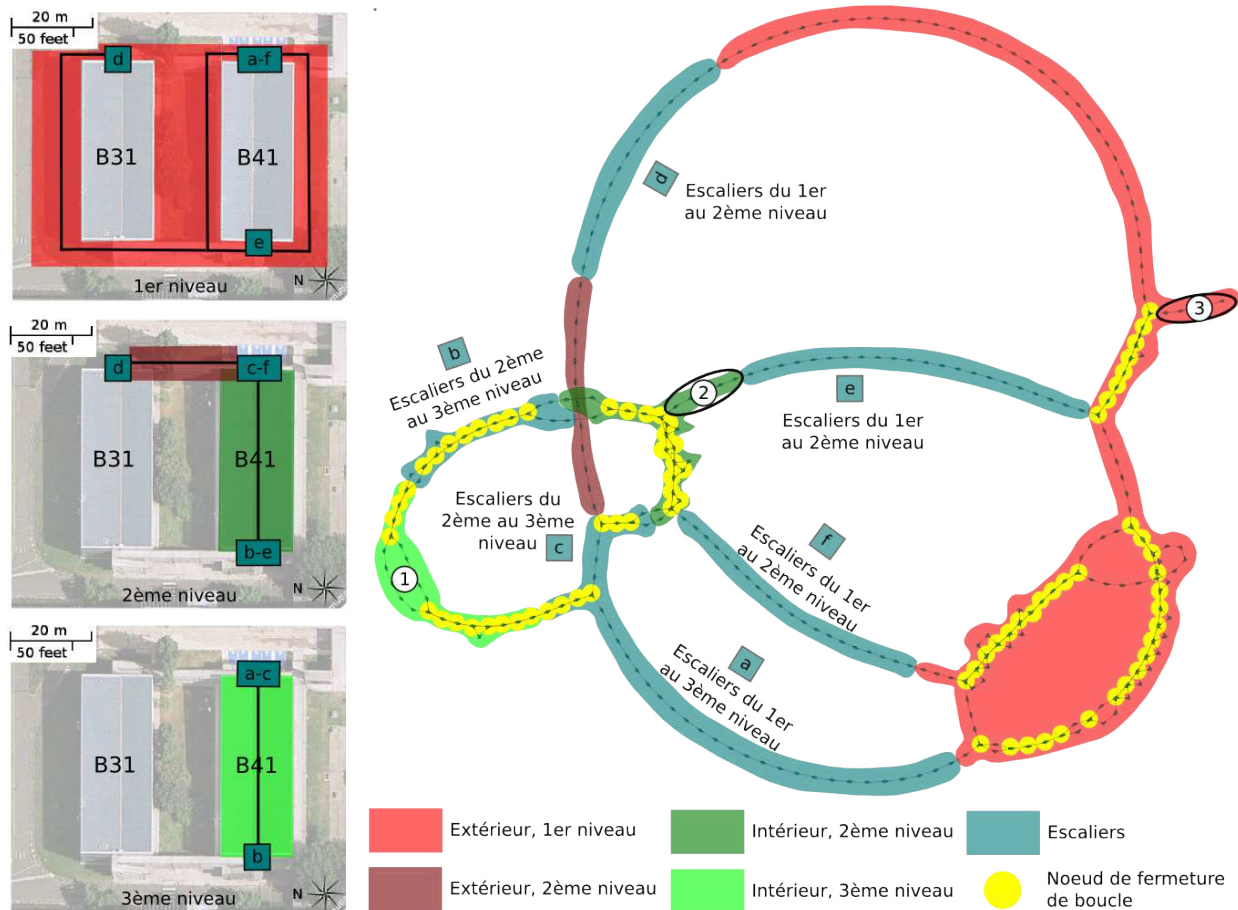


FIG. 7.6: Plan sur plusieurs niveaux de l'environnement parcouru (partie gauche de la figure) et carte topologique correspondante (partie droite de la figure). La disposition du graphe est réalisée grâce à un simple algorithme de relaxation [Kamada and Kawai, 1989]. Les détails concernant la trajectoire et la carte sont donnés dans le texte.



FIG. 7.7: Exemples d'images provenant des deuxième (rangée du haut) et troisième (rangée du bas) niveaux. On se rend compte de la similarité entre les images.

Il y a deux points qui méritent d'être détaillés en analysant les résultats de cette expérience. Premièrement, aucun faux positif n'a été détecté : cela sous-entend que l'algorithme de détection de fermeture de boucle n'a jamais proclamé qu'une image venait d'un lieu connu si ce n'était pas le cas en réalité. Comme nous l'avons déjà souligné dans les résultats expérimentaux de la partie I, ceci revêt une importance capitale pour assurer la viabilité de l'estimation. Cette performance est rendue d'autant plus difficile ici au vu de l'important aliasing perceptuel présent dans cet environnement composé à la fois de séquences d'intérieur et d'extérieur. La seconde constatation qui peut être faite concerne encore une fois la faible réactivité du modèle de probabilité, particulièrement visible dans la carte de la figure 7.6 où certaines portions de l'environnement sont parcourues plusieurs fois et ne présentent cependant pas de noeuds de fermeture de boucle (deux de ces portions sont encadrées en noir dans le graphe et annotées avec les numéros "2" et "3").

### 7.2.1 Influence des espaces de représentation

Nous avons déjà pu observer les effets bénéfiques de la combinaison des espaces de représentation dans le chapitre 3 de la partie I. Afin d'en percevoir les conséquences sur le processus de construction de carte topologique, nous présentons ici une analyse comparative des résultats en fonction des caractérisations retenues pour les images. La figure 7.8 illustre sur cette séquence le gain en réactivité déjà remarqué dans le chapitre 3 de la partie I lorsque la représentation mixte (i.e., primitives SIFT et histogrammes H) est employée.

En considérant plusieurs caractérisations de la même image, il est possible de tirer profit de l'information de structure fournie par les primitives SIFT, et de l'information de couleur enregistrée dans les histogrammes de teinte. Cela s'est montré particulièrement efficace dans les parties de la séquence qui ont été acquises en intérieur. En effet, comme on peut s'en apercevoir d'après les images de la figure 7.7, l'environnement

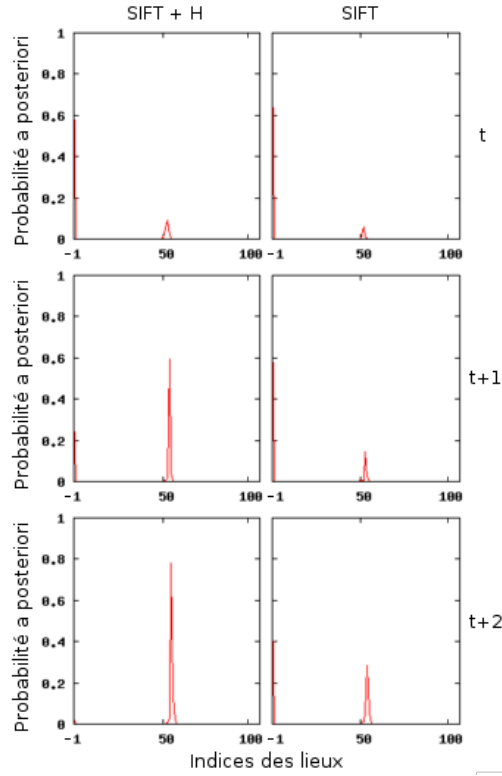


FIG. 7.8: *Espaces de représentation.* Chaque colonne de la figure donne l'évolution au cours du temps de la probabilité de détection de fermeture de boucle. La colonne de gauche donne cette évolution quand les primitives SIFT sont combinées avec les histogrammes  $H$ , alors que dans la colonne de droite, cette même évolution est donnée pour les primitives SIFT utilisées seules. Comme on peut le remarquer, le seuil fixé pour la probabilité a posteriori (i.e., 0.8) est atteint plus rapidement dans le premier cas (i.e., dans le second cas deux images supplémentaires ont été nécessaires pour que le seuil soit atteint).

d'intérieur est composé de murs sans texture, où les primitives SIFT sont peu nombreuses et faiblement discriminantes. Par ailleurs, étant donné que les couloirs de cet environnement d'intérieur sont principalement composés de trois couleurs (i.e., blanc sur les murs, rouge et gris foncé au sol), l'information de teinte paraît plus pertinente. De ce fait, la prise en compte des deux espaces de représentation permet d'améliorer les performances de détection de fermeture de boucle, comme illustré dans la figure 7.9. Dans cette figure, la portion de la carte correspondant aux images d'intérieur du 3<sup>ème</sup> étage et obtenue en utilisant les primitives SIFT uniquement est comparée à cette même portion lorsque les histogrammes  $H$  sont également employés : le nombre de noeuds de fermetures de boucles est bien plus faible dans le premier cas, et la carte correspondante est en conséquence moins cohérente.



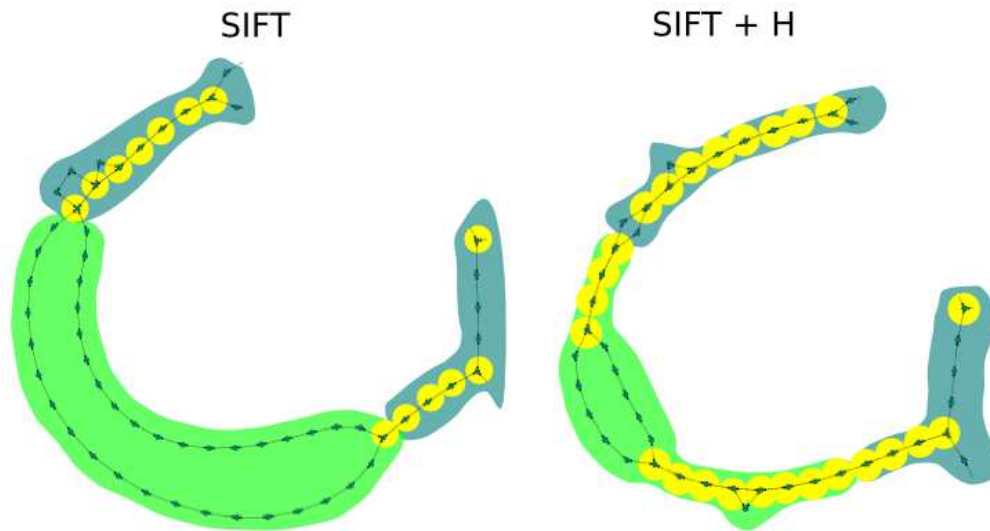


FIG. 7.9: *Espaces de représentation (2). La figure donne la portion de la carte qui correspond à l'environnement d'intérieur du 3<sup>ème</sup> étage (aux alentours du cercle "1" dans la figure 7.6), lorsque les primitives SIFT sont utilisées seules (gauche) ou en combinaison avec les histogrammes de teinte (droite). Comme on peut le remarquer, moins de fermetures de boucles sont détectées lorsque les histogrammes *H* ne sont pas pris en compte (i.e., le nombre de noeuds de fermeture de boucle est plus faible). La carte résultante est de fait moins cohérente (i.e., la plupart des noeuds sont dupliqués). Le code couleur employé ici est le même que celui de la figure 7.6.*

### 7.2.2 Performances

Dans la séquence mixte traitée ici, l'importante profondeur de champ des scènes d'intérieur et d'extérieur a permis de choisir une fréquence d'acquisition d'images de 0.5Hz. Ainsi, pour atteindre des performances en temps réel, le temps total de traitement d'une image doit être effectué en moins de 2s. Comme le montre l'évolution des temps de calcul donnée dans la figure 7.10, cette limite supérieure n'est jamais atteinte. Dans la figure 7.10, les temps de calcul sont donnés pour tous les espaces de représentation confondus. On peut notamment remarquer que le temps d'extraction des primitives est borné entre 250ms et 500ms. Comme cela a déjà été observé dans les résultats expérimentaux de la partie I, le temps total des traitements semble évoluer de façon linéaire au cours du temps, au moins jusqu'à la 200<sup>ème</sup> image (i.e., aux alentours du 40<sup>ème</sup> indice dans le graphe) : à partir de là, l'évolution stagne, diminuant même un peu sur la fin. Ceci est certainement dû à la nature cyclique de la trajectoire de la caméra : lorsque la caméra visite à nouveau des zones déjà cartographiées, peu de nouveaux mots sont ajoutés au dictionnaire, et moins d'hypothèses sont ajoutées au modèle de l'environnement. En conséquence, les temps de calcul sont plus faibles.

Toutefois, en raison de la taille relativement importante des images acquises ici (i.e., 320x240 contre 240x192 pixels dans les expériences de la partie I), les traitements de l'image sont lourds, générant plusieurs centaines de primitives qu'il faut apparier avec le dictionnaire et résultant au final en des temps de

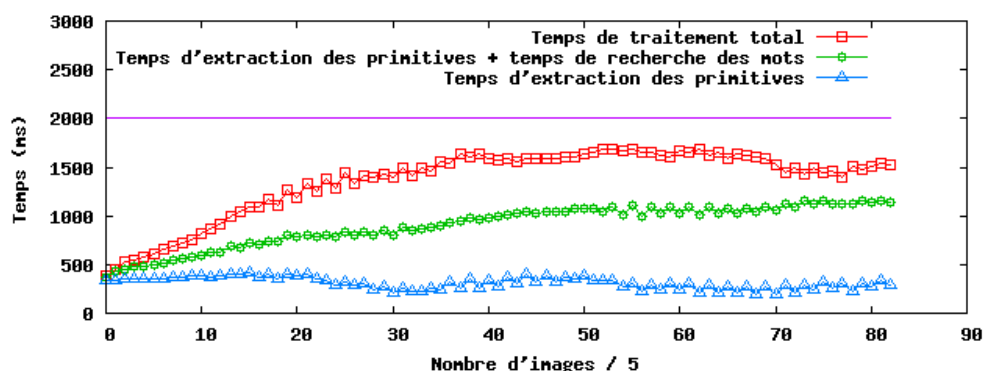


FIG. 7.10: Évolution des temps de traitements par image : la figure donne le temps requis pour extraire les primitives dans les images (triangles), auquel est ajouté le temps nécessaire à la recherche des mots correspondants dans le vocabulaire (cercles), avec enfin le temps total de traitement par image (carrés). Pour améliorer la lisibilité, les temps de calcul ont été moyennés toutes les 5 images.

calcul proches de la limite supérieure. L'augmentation de la complexité des traitements est visible dans la figure 7.11 donnant l'évolution de la taille du vocabulaire SIFT au cours du temps pour deux différentes taille d'images : l'évolution est d'autant plus rapide que l'image est grande. Finalement, les performances pourraient être sensiblement améliorées par l'emploi de détecteurs de primitives optimisés pour être plus rapides.

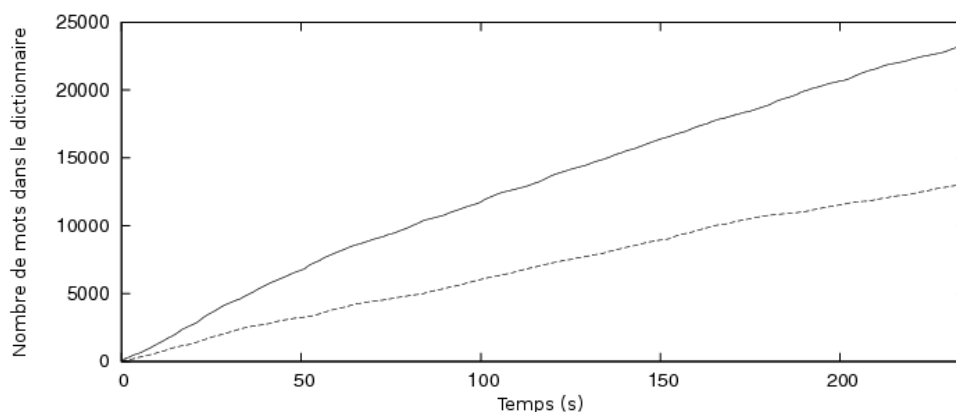


FIG. 7.11: Évolution au cours du temps de la taille du vocabulaire SIFT : la courbe en trait plein correspond à une taille d'images de 320x240 pixels, alors que la courbe en tirets correspond à une taille de 240x192 pixels.

Le tableau 7.2 donne des informations supplémentaires sur les traitements : il recense la longueur de la séquence et le nombre d'images qu'elle contient, le temps total CPU requis pour traiter la séquence, ainsi que les tailles des dictionnaires SIFT et histogrammes H à la fin de l'expérience. D'après ce tableau, on

TAB. 7.2: *Performances.*

Longueur	#img	CPU	#SIFT	#Hist. H	#FB	%VP	#FA
13m50s	415	9m17s	79243	6864	184	63	14

s'aperçoit que la taille des vocabulaires est importante (notamment dans le cas des primitives SIFT) au vu des résultats présentés dans [Cummins and Newman, 2007]. Nous avons déjà évoqué cette particularité dans le chapitre 3 de la partie I : ceci est dû au pouvoir discriminant accordé aux mots. Le tableau 7.2 contient aussi des informations au sujet des performances en termes de reconnaissance, en donnant notamment le nombre de fermetures de boucles dans la séquence (“#FB”, déterminé à la main d’après la trajectoire de la caméra), le taux de vrais positifs (“%VP”, les fermetures de boucles correctement détectées), et le nombre de fausses alarmes (“#FA”, hypothèses erronées qui reçoivent une probabilité élevée mais qui sont écartées par l’algorithme de géométrie multi-vues). Les fausses alarmes apparaissent surtout au début de la séquence, alors que peu de statistiques ont été recueillies pour les mots afin d’en déduire un coefficient tf-idf efficace. Ce point a déjà été abordé dans la discussion de la partie I (chapitre 4).

Lorsque l’on considère le nombre de fermetures de boucles existant dans la séquence (i.e., 184), il est important de préciser que 37 d’entre elles correspondent à des passages en des lieux connus avec un changement de point de vue de  $180^\circ$  : la caméra est effectivement au même endroit que précédemment, mais avec un point de vue complètement opposé. Dans ces conditions, il semble quasiment impossible de détecter les fermetures de boucles correspondantes avec une simple caméra monoculaire. Ainsi, sans prendre en compte ces 37 images, le taux de détections correctes de fermeture de boucle est de 78%.

### 7.3 Prise en compte de l’information d’odométrie

Dans cette dernière section, nous présentons des résultats préliminaires concernant l’ajout d’information métrique dans la carte. Pour cela, nous avons déplacé, dans un environnement d’intérieur, un robot équipé d’une caméra et fournissant des mesures d’odométrie, en veillant à ce que la trajectoire suivie présente plusieurs cycles. La plateforme utilisée pour ces expériences est le Pioneer 3-DX de ActivMedia Robotics (cf. figure 7.12). Ce robot est composé de :

- un PC embarqué
- deux moteurs / codeurs permettant de déplacer le robot avec précision
- un télémètre laser fournissant une vue de l’environnement à  $180^\circ$  avec un pas d’un degré
- 16 sonars
- une caméra couleur
- une liaison WiFi

La figure 7.13 donne un aperçu de l’amélioration apportée par l’algorithme de relaxation sur le processus d’inférence de la position des noeuds. Comme on peut le constater, la carte construite avant la relaxation est



FIG. 7.12: *Le Pioneer P3-DX de ActivMedia Robotics.*

incohérente : beaucoup de noeuds se retrouvent en dehors du plan du bâtiment dans lequel l'expérience a été conduite. Après la relaxation, les positions estimées sont beaucoup plus cohérentes, et la carte semble bien représenter la topologie de l'environnement parcouru. On remarque cependant quelques imprécisions qui placent encore certains noeuds en dehors du plan (i.e., dans le bas du plan, le long du couloir), mais avec une erreur de position d'un ordre de grandeur beaucoup moins important que lorsque la relaxation n'est pas appliquée.

La figure 7.14 donne un autre exemple de carte obtenue pour une expérience de plus longue durée, avec plusieurs fermetures de cycles de différentes tailles, montrant ici aussi une forte amélioration de la cohérence des positions des noeuds.

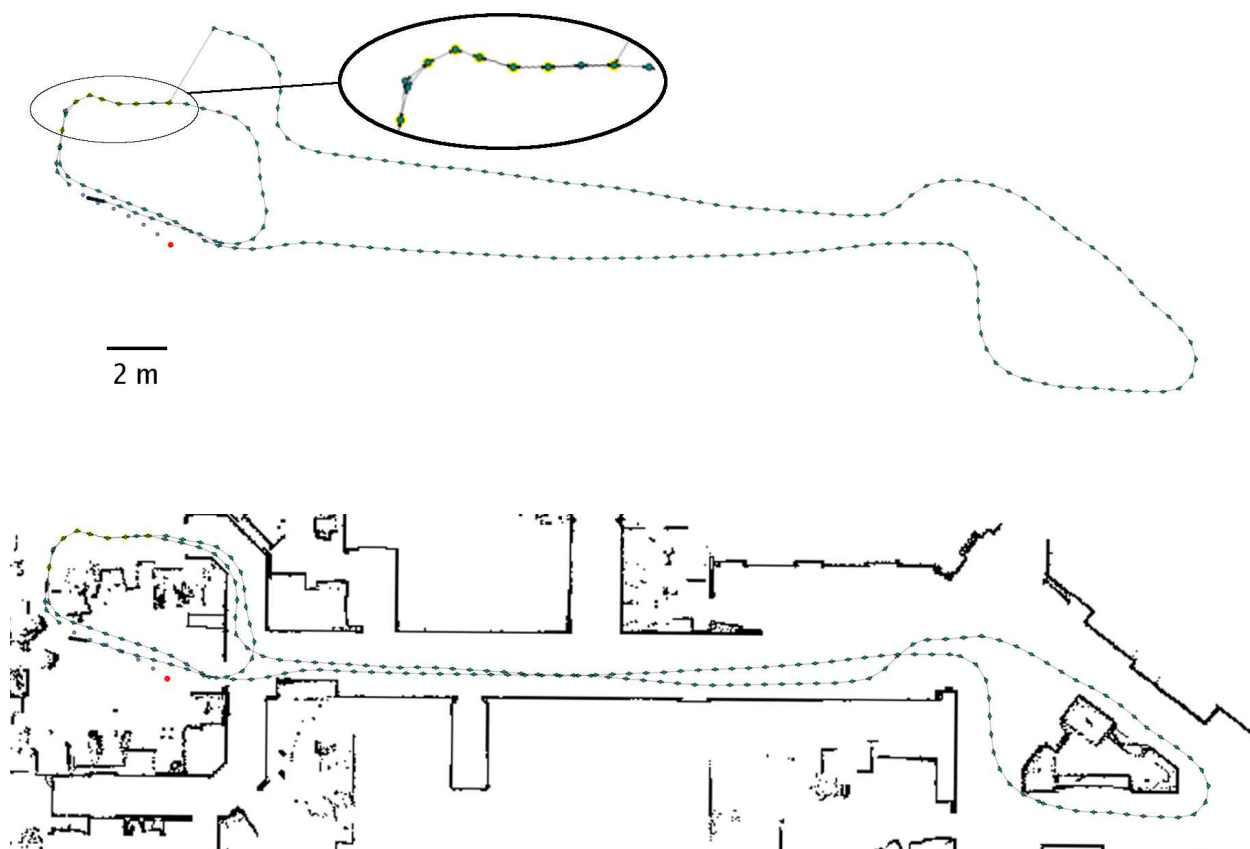


FIG. 7.13: Carte topologique avant (partie du haut) et après (partie du bas) la relaxation. Dans le premier cas, alors qu'une fermeture de boucle est détectée, la carte est incohérente. Dans le second cas, la carte topologique est superposée sur un plan du bâtiment avec la bonne échelle. On peut notamment observer que les positions estimées pour les noeuds sont plus cohérente et la topologie de l'environnement est respectée.

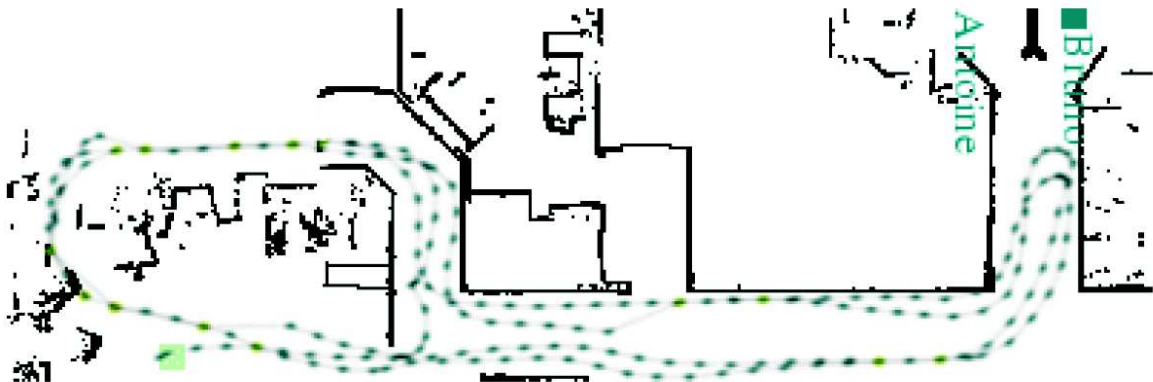


FIG. 7.14: Carte topologique obtenue après relaxation dans le cas d'une trajectoire assez longue avec plusieurs fermetures de boucles.

# Chapitre 8

## Discussion

L’approche basée sur l’apparence pour le SLAM topologique proposée ici peut être comparée favorablement avec l’ensemble des méthodes citées dans l’état de l’art de cette partie (cf. page 95), car notre solution est la seule à combiner performances en temps réel et traitements incrémentiels.

Les résultats obtenus ici confirment la robustesse de notre méthode de détection de fermeture de boucle en permettant la construction de cartes topologiques valides uniquement sur la base d’une caméra monoculaire grand-angle : le modèle de sacs de mots visuels développé par [Filliat, 2007] a permis d’obtenir de bonnes performances de reconnaissance de lieu, sans avoir à “aplatir” les images pour en retirer la distorsion radiale. En conséquence, il a été possible de détecter les fermetures de boucles simplement même sans utiliser une caméra à projection perspective standard.

Dans l’état de l’art dressé dans cette partie, nous avons mentionné deux solutions ([Cummins and Newman, 2007] et [Fraundorfer et al., 2007]) au problème du SLAM topologique qui sont basées sur l’apparence uniquement et qui reposent sur une caractérisation des images provenant d’une caméra monoculaire selon le formalisme des sacs de mots visuels : ces deux solutions se placent donc dans un cadre très proche du notre. La première caractéristique majeure qui oppose notre approche à ces méthodes tient dans l’aspect incrémentiel des traitements : dans [Cummins and Newman, 2007] et [Fraundorfer et al., 2007] en effet, une phase préalable hors-ligne est nécessaire pour l’apprentissage du dictionnaire. Pour le reste, il semble que notre modèle de vraisemblance soit similaire à celui mis en oeuvre dans [Fraundorfer et al., 2007], étant donné qu’une méthode de vote y est employée pour prédire le lieu de l’image courante, avec également une vérification ultérieure par un algorithme de géométrie multi-vues. Cependant, notre méthode propose un cadre plus robuste pour l’inférence du lieu de provenance de chaque image, puisque nous utilisons un modèle de probabilité établi dans un cadre de filtrage Bayésien pour déterminer ce lieu. Les auteurs de [Fraundorfer et al., 2007] se basent quant à eux simplement sur le critère du maximum de vraisemblance pour prendre cette décision, et nous avons pointé les faiblesses de ce critère de décision dans le chapitre 1 de la partie I (section 1.4.2). Toutefois, comme cela a déjà été remarqué dans la discussion de la partie I, le

modèle de probabilité appliqué dans [Cummins and Newman, 2007] semble présenter une robustesse accrue face au problème de l'aliasing perceptuel. Enfin, en termes de performances, d'après les résultats présentés dans [Fraundorfer et al., 2007] les traitements sont réalisés en temps réel, comme dans notre solution, alors que la version accélérée [Cummins and Newman, 2008b] de [Cummins and Newman, 2007] ne permet pas encore d'atteindre ce genre de performance. Ainsi, il apparaît que notre solution se place dans une position intermédiaire par rapport aux travaux de [Fraundorfer et al., 2007] et de [Cummins and Newman, 2007] : le modèle de probabilité que nous avons développé offre une meilleure robustesse que le critère retenu dans [Fraundorfer et al., 2007], sans pour autant atteindre le niveau de [Cummins and Newman, 2007], mais en ayant toutefois l'avantage de proposer des traitements réalisables en temps réel et surtout, de manière complètement incrémentielle.

## 8.1 Perspectives

Dans la présentation de notre solution et des résultats expérimentaux correspondant, nous avons mentionné des travaux préliminaires concernant l'ajout d'information métrique dans la carte topologique de l'environnement. Les expériences conduites dans ce domaine, même si elles correspondent à des développements encore en cours, ont permis de démontrer l'intérêt d'utiliser une telle information dans le cadre du SLAM topologique. Ainsi, les perspectives d'évolution de l'approche détaillée dans cette partie concernent essentiellement cet aspect.

### 8.1.1 Limites de la vision monoculaire

Malgré la bonne qualité des résultats obtenus ici, la vision monoculaire n'est pas adaptée lorsque les variations dans les changements de points de vue sont importantes entre les différents passages de la caméra au même endroit. Pour outrepasser cette difficulté, il faut ajouter une information métrique sur les arêtes de la carte. Ainsi, à partir de cette information, on pourrait détecter les fermetures de boucles pour lesquelles la caméra est proche d'un lieu connu, mais avec une orientation qui l'empêche de reconnaître correctement ce lieu.

L'ajout d'une information métrique sur les arêtes de la carte peut être envisagé de deux manières différentes. Il est par exemple possible de reposer pour cela sur une estimation locale de la position et de l'orientation de la caméra, grâce à une odométrie visuelle [Nistér et al., 2004] ou bien même à partir d'un algorithme de SLAM visuel tel que celui présenté dans [Davison et al., 2007]. C'est notamment la solution retenue par les auteurs de [Kim and Kweon, 2007], [Konolige and Agrawal, 2007] et [Steder et al., 2007]. Cependant, ces approches basées purement sur la vision pour l'estimation de la position et de l'orientation de la caméra dépendent fortement d'un suivi image par image précis des primitives au cours du temps. Ainsi, elles peuvent rapidement être mises en défaut lorsque l'information visuelle n'est pas utilisable (par exemple en cas d'obstruction ou de dysfonctionnement de la caméra, ou bien lorsque le suivi est interrompu par le



manque de structure dans l'environnement). Dans une perspective plus expérimentale, l'information métrique locale peut être obtenue en montant la caméra sur un robot qui fournit des mesures d'odométrie. Cela permet alors de détecter les fermetures de boucles même lorsque l'information visuelle n'est pas utilisable. C'est la solution choisie dans les travaux de [Hubner and Mallot, 2007] et de [Rybski et al., 2003].

### 8.1.2 Navigation

Il apparaît donc indispensable d'ajouter une information métrique à la carte. Non seulement cela permettrait d'améliorer le taux de détections correctes de fermetures de boucles, mais cette information pourrait également être utilisée à des fins de navigation : pour la planification d'un chemin dans la carte, l'information métrique locale encodée dans les arêtes permet de déterminer le chemin à suivre pour joindre les noeuds voisins. Toutefois, il serait également possible de mettre en place un algorithme de *homing* ou d'asservissement visuel basé uniquement sur l'apparence, reposant sur une information qualitative de direction (comme dans [Filliat, 2008] ou [Remazeilles and Chaumette, 2007]) pour enregistrer les chemins à suivre pour transiter entre lieux adjacents.

### 8.1.3 Fusion des noeuds

Nous avons beaucoup insisté, dans cette partie et dans la précédente, sur la faible réactivité du modèle de probabilité, entraînant dans certains cas la duplication d'une partie de la carte. Afin d'obtenir un modèle plus viable, on pourrait envisager dans le cadre de l'application au SLAM topologique de combiner des noeuds voisins sur la base de la géométrie multi-vues : si un noeud est directement lié à un noeud de fermeture de boucle, on peut essayer de trouver une transformation exprimant un changement de point de vue valide entre les lieux correspondants. En cas de succès, les noeuds peuvent être fusionnés, résultant au final en une carte plus cohérente. Dans le même état d'esprit, l'ajout d'information métrique provenant de l'odométrie (comme proposé plus haut) permettrait d'améliorer encore la cohérence de la carte. En effet, en disposant d'une position métrique précise pour les noeuds, on pourrait fusionner des noeuds non directement reliés par une arête, mais dont les positions sont proches. Il faudrait tout de même veiller à ce que les noeuds considérés possèdent une apparence suffisamment similaire, en comptant le pourcentage de mots communs qu'ils contiennent par exemple, mais il faudrait également vérifier que la contrainte de géométrie épipolaire puisse être validée entre leurs points de vue.

### 8.1.4 Modèle d'évolution temporelle

Enfin, une information métrique de déplacement du robot entre deux acquisitions consécutives d'images permettrait également d'améliorer le modèle d'évolution temporelle mis en place dans le cadre de l'estimation de la probabilité de fermeture de boucle (ce modèle est décrit dans la section 2.3.1 du chapitre 2 de la partie I). En effet, la version actuelle de ce modèle est basée sur la présence de similarités entre les

images consécutives. Ainsi, lorsque le robot se trouve en un noeud relié à plusieurs voisins, la probabilité de transition est uniformément répartie entre ces différents voisins. Grâce à l'information de déplacement du robot obtenue par odométrie, on pourrait cibler la propagation sur les noeuds se trouvant dans la direction du robot, les autres étant moins pertinents.

## 8.2 Conclusion

La solution proposée ici pour le SLAM topologique résulte d'une application directe de la méthode de détection de fermeture de boucle présentée dans la partie I. Ainsi, notre méthode permet la construction de cartes topologiques de l'environnement de manière incrémentielle, au fur et à mesure que les images sont acquises par le robot, et sans informations a priori sur le type d'environnement. Pour cela, l'algorithme développé repose sur une simple caméra monoculaire dont les images d'entrée peuvent être caractérisées dans deux espaces de représentation distincts pour caractériser les noeuds de la carte, alors que les arêtes liant ces noeuds encodent une simple information d'adjacence temporelle.

Encore une fois, les contraintes énumérées en introduction de ce mémoire sont ici pleinement satisfaites, puisque le modèle de l'environnement est construit en-ligne grâce à des traitements réalisables en temps réel. D'autre part, les cartes obtenues sur la base de l'apparence uniquement sont cohérentes avec la nature cyclique de la trajectoire suivie par le robot lors de son déplacement, et ce même en présence d'un aliasing perceptuel important.

## **Troisième partie**

# **Application au SLAM métrique**



## Chapitre 9

# État de l’art

La dernière partie de ce mémoire présente une autre application de la solution de détection de fermeture de boucle détaillée dans la partie I. Ici, nous considérons précisément un des cas d’utilisation exposés dans l’introduction de ce mémoire pour lequel la détection de fermeture de boucle permet d’améliorer grandement la qualité du processus d’estimation : il s’agit d’une adaptation de notre solution pour récupérer d’un kidnapping dans une application de SLAM métrique. La détection de fermeture de boucle permet de retrouver une position et une orientation valides pour la caméra après avoir arbitrairement déplacé celle-ci dans une partie connue de l’environnement, mais sans connaissances sur le mouvement effectué. L’algorithme de SLAM métrique choisi pour cette expérience est présenté dans [Davison, 2003] et [Davison et al., 2004] : il s’agit d’une méthode basée sur le *filtre de Kalman étendu* qui permet d’inférer la pose<sup>1</sup> et les vitesses de translation et de rotation de la caméra, ainsi que les positions 3D d’amers ponctuels de l’environnement. Pour réaliser cette estimation, l’algorithme *MonoSLAM* ne requiert qu’une simple caméra monoculaire, en traitant en temps réel les images qu’elle renvoie à 30Hz.

L’objectif de cet état de l’art est de proposer un aperçu des méthodes basées vision qui permettent d’aborder le problème de la détection de fermeture de boucle ainsi que celui de la mise à jour correspondante dans le processus d’estimation du SLAM métrique. Pour commencer, nous dressons une revue générale des méthodes de SLAM métrique, en exposant brièvement les différentes techniques de filtrage et de représentation de l’environnement les plus couramment utilisées. Ensuite, nous nous concentrons sur le sujet de la fermeture de boucle pour le SLAM métrique, en présentant différentes solutions et en les regroupant en fonction du cadre d’estimation dans lequel elles se situent. Enfin, une conclusion clôt ce chapitre.

---

<sup>1</sup>Dans la suite de ce mémoire, le terme “pose” sera employé pour désigner la position orientée (i.e., position et orientation) de la caméra.

## 9.1 Revue générale

Dans l'historique des solutions proposées au problème du SLAM métrique, on peut certainement retenir les travaux de [Smith et al., 1987] comme étant la base de toute la famille de méthodes qui ont émergé par la suite : les auteurs de [Smith et al., 1987] ont en effet été les premiers à saisir et à formaliser concrètement le caractère simultané de l'estimation de la position du robot et des amers de la carte. Comme nous le mentionnons un peu plus loin, les approches métriques au problème du SLAM nécessitent une fusion d'informations provenant des capteurs et des effecteurs du robot. Pour cela, plusieurs techniques de filtrage ont été développées, chacune présentant des caractéristiques de complexité et d'approximation différentes. De plus, on peut choisir de construire une carte métrique composée de simples amers ponctuels ou bien contenant des objets géométriques plus complexes. Ainsi, il n'existe pas à ce jour de solution standard au problème du SLAM métrique, mais plutôt un ensemble d'approches et de représentations parmi lesquelles il faut faire un choix en fonction du type d'environnement, des caractéristiques de mobilité du robot, et des capacités des capteurs embarqués.

### Différents types de représentations

Dans le cadre des approches métriques au SLAM, l'environnement est représenté par une collection d'amers localisés précisément (i.e., par une position géométrique) dans un référentiel. Une position géométrique pour le robot est elle aussi inférée dans ce repère, en fusionnant généralement deux sources d'information : les données relatives au déplacement du robot entre deux observations (données obtenues par l'odométrie par exemple), ainsi que les positions des amers telles que perçues depuis la position courante. Les amers sélectionnés peuvent être de différente nature, en fonction essentiellement du type de capteur embarqué à bord du robot. Ainsi, les capteurs de distance tels que les télémètres laser permettent de modéliser l'environnement par une carte d'amers ponctuels, sous la forme de nuages de points disséminés à la surface des objets ([Lu and Milios, 1997]), mais ils permettent également de prendre en compte des informations de plus haut niveau pour la représentation, comme par exemple des polygones ([Dufourd, 2005]). La vision permet elle aussi une caractérisation de l'environnement sous la forme d'amers ponctuels ([Davison et al., 2004]) ou d'objets plus complexes, comme dans les travaux de [Smith et al., 2006] où les auteurs utilisent des segments pour la construction de la carte. Enfin, il est possible dans le cadre des approches métriques de rendre compte explicitement du caractère occupé ou libre de l'espace, comme par exemple dans les grilles d'occupation (cf. section 1.2.1 du chapitre 1 de la partie I).

### Différentes méthodes pour le filtrage

Originellement ([Smith et al., 1987]), la solution proposée pour fusionner les informations relatives au déplacement du robot avec celles provenant de l'observation des positions des amers reposait sur l'emploi d'un *filtre de Kalman étendu (FKE)*. Dans le FKE, les quantités estimées (i.e., les positions du robot et

des amers de l'environnement) sont concaténées dans un *vecteur d'état*, auquel on associe une matrice de covariance renseignant sur l'incertitude liée à l'estimation de ces quantités. C'est par le biais de cette matrice de covariance que sont maintenues les relations entre les quantités estimées. Le FKE présente trois inconvénients. Pour commencer, il offre une complexité quadratique en la taille du vecteur d'état (et donc en le nombre d'amers contenus dans la carte). Deuxièmement, ce n'est pas un estimateur optimal lorsque les équations régissant les modèles de déplacement et d'observation du robot sont non-linéaires. Enfin, son application est restreinte à des cas d'utilisation où les bruits sur ces différents modèles sont Gaussiens. En dépit de ces inconvénients, le FKE est aujourd'hui encore très largement utilisé ([Dissanayake et al., 2001]), même dans le cadre de la vision ([Davison et al., 2007]).

Dans sa version à "états retardés" (i.e., *delayed state*), le FKE est utilisé pour estimer la trajectoire complète du robot, et non uniquement sa position courante. Une carte n'est pas estimée ici, les positions des amers étant exprimées localement aux différentes positions auxquelles ils ont été perçus le long de la trajectoire. Dans cette version, le FKE offre notamment la possibilité de gérer facilement les événements de fermeture de boucle, comme en attestent les travaux de [Eustice et al., 2004]. Toutefois, en raison de sa complexité quadratique en la taille du vecteur d'état (i.e., la longueur de la trajectoire ici), son application reste limitée à des trajets assez courts.

D'autres méthodes proposent une estimation de la trajectoire complète du robot plutôt que de la position courante uniquement. Cependant, le processus d'estimation sous-jacent ne repose plus sur un filtre de Kalman étendu, mais sur un algorithme de relaxation : ce dernier cherche à satisfaire les contraintes régissant la position d'équilibre d'un réseau de ressorts. Développé au départ dans le cadre des capteurs de distance ([Lu and Milios, 1997]), cette approche a également été adaptée à la vision ([Konolige and Agrawal, 2007]). Le principal avantage de ces méthodes à base de graphe émane de la complexité constante d'une intégration de mesure provenant des capteurs.

Parmi les méthodes proposant une estimation de l'historique de la trajectoire du robot, les approches de filtrage particulaire (comme par exemple FastSLAM, [Montemerlo et al., 2002]) ont l'avantage de pouvoir gérer plusieurs hypothèses au cours du temps. Elles offrent ainsi la possibilité de prendre en compte explicitement les associations de données ambiguës, ce qui leur permet d'assurer une estimation viable même en cas d'aliasing perceptuel. Malgré tout, certaines caractéristiques du processus de filtrage en limitent la pertinence lors de fermetures de boucles (voir section 9.3 de cet état de l'art).

Une alternative au filtrage particulaire permettant la gestion d'hypothèses multiples revient à utiliser un banc de filtres de Kalman étendus, comme proposé par les auteurs de [Jensfelt and Kristensen, 1999]. Cette solution présente l'avantage d'approximer la distribution de probabilité de la position du robot dans la carte de manière continue, sous la forme d'une mixture de Gaussiennes, alors que dans le cadre du filtrage particulaire, cette approximation est réalisée de manière discrète, par le biais d'un ensemble d'échantillons.

Enfin, on peut mentionner la variante "filtre d'information" du filtre de Kalman ([Thrun and Montemerlo, 2006], [Eustice et al., 2005]) : en inversant la matrice de covariance, on obtient une matrice d'infor-

mation pour laquelle les étapes de mise à jour ne requièrent que des opérations de complexité constante. Toutefois, pour l'association de données il faut reconstruire le vecteur d'état à partir du vecteur d'information, ce qui est une opération coûteuse. Il existe cependant des méthodes permettant de maintenir une estimation du vecteur d'état au cours du temps en parallèle du vecteur d'information, sans avoir à effectuer explicitement la reconstruction : on transcrit pour cela les mises à jour relatives au vecteur d'information en mises à jour pour le vecteur d'état, en réalisant quelques approximations ([Thrun et al., 2005]).

### **Caractéristiques générales**

Le principal avantage des cartes métriques est de pouvoir représenter l'environnement de manière plus exhaustive que dans le cas des approches topologiques où seulement les lieux visités sont explicitement modélisés : sur la base des représentations métriques, il est en effet possible de planifier un déplacement dans des zones encore inexplorées. A partir d'une carte métrique, on peut essayer d'extraire des caractéristiques de plus haut niveau pour la représentation de l'environnement, en regroupant par exemple des amers proches pour tenter d'inférer les délimitations des objets. Toutefois, pour obtenir une carte fidèle, des modèles précis des capteurs (i.e., pour les observations) et des effecteurs (i.e., pour les déplacements) du robot sont indispensables. Par ailleurs, le calcul d'un chemin dans une représentation métrique est généralement complexe, en raison de la nature continue de l'espace dans lequel ce chemin est recherché.

Le lecteur intéressé pourra trouver une présentation plus détaillée et un historique plus complet au sujet des approches métriques dans [Filliat, 2001], [Filliat and Meyer, 2003], [Meyer and Filliat, 2003], [Thrun, 2002] ou encore [Thrun et al., 2005].

## **9.2 Détection de fermeture de boucle dans le cadre du filtre de Kalman étendu**

La deuxième section de cet état de l'art concerne les techniques récentes d'association de données basées sur la vision qui permettent d'améliorer les performances du populaire filtre de Kalman étendu avec des capacités avancées de détection de fermeture de boucle. Par exemple, les auteurs de [Lemaire et al., 2007] proposent une méthode simple pour la détection de fermeture de boucle qui repose sur l'estimation de la position. Les amers de la carte dont le point de vue est à une certaine distance de la position actuellement estimée pour la caméra sont régulièrement appariés avec l'image courante. Si un nombre significatif de correspondances est trouvé, la localisation géométrique de la caméra est mise à jour en fonction des positions de ces amers. Ces travaux ont par ailleurs été adaptés [Lemaire and Lacroix, 2007] au cadre de la vision panoramique. Dans un contexte similaire, les auteurs de [Frintrop and Cremers, 2007] reposent sur une approche semblable pour la détection de fermeture de boucle, en employant toutefois un mécanisme d'attention visuelle pour prédire l'occurrence d'amers de la carte dans l'image courante. La solution propo-



sée implémente une stratégie “descendante” pour faire cette prédiction (voir section 1.3.1 du chapitre 1 de la partie I).

Dans les approches mentionnées ci-dessus, c’est la simple observation d’amers de la carte perçus initialement lors d’un passage antérieur en un lieu donné qui permet de retrouver une localisation correcte pour la caméra. La solution développée dans [Clemente et al., 2007] et [Se et al., 2005] considère ce problème d’une manière différente : plutôt que de reconnaître simplement d’anciens amers dans l’image courante, les auteurs proposent de segmenter l’environnement en sous-cartes d’amers ponctuels, abordant alors la détection de fermeture de boucle comme un problème de détection de recouvrement entre sous-cartes distantes. Si un recouvrement plausible est trouvé, un algorithme de relaxation est utilisé pour aligner les sous-cartes et en déduire une localisation cohérente. Pour améliorer la robustesse de la procédure de détection de recouvrement, les auteurs de [Clemente et al., 2007] emploient l’algorithme *GCBB* (*Geometric Constraints Branch and Bound*, [Neira et al., 2003]), basé sur le *JCT* (*Joint Compatibility Test*, [Neira and Tardós, 2001]) pour l’association de données. Ce système repose à la fois sur la similarité dans l’apparence visuelle (établie sous la forme de contraintes unaires), et sur les distances relatives entre amers (correspondant à des contraintes binaires), afin de trouver l’ensemble le plus important possible d’amers communs à deux sous-cartes.

### 9.2.1 Introduction d’une nouvelle estimation de pose dans le FKE

Les solutions présentées jusque-là dans cet état de l’art reposent sur l’observation d’anciens amers pour le recalage de la pose de la caméra : en reconnaissant des zones de l’environnement explorées par le passé et enregistrées dans la carte, le FKE permet de rétablir naturellement l’estimation de la pose de la caméra à l’endroit correspondant. Les auteurs de [Williams et al., 2007b] proposent une solution alternative à ce problème, en introduisant dans le FKE une pose calculée par ailleurs en cas de fermeture de boucle. Leur méthode permet de rétablir la pose de la caméra même lorsque le suivi de son estimation est perdu (après un mouvement rapide ou une obstruction de la caméra par exemple). Pour cela, une procédure RANSAC [Fischler and Bolles, 1981] efficace permet de retrouver rapidement les amers de la carte qui correspondent aux primitives visuelles actuellement perçues et à partir desquelles une localisation cohérente peut être inférée en temps réel [Williams et al., 2007a]. Cette inférence est réalisée grâce à l’algorithme des “trois points” [Fischler and Bolles, 1981] : plusieurs triplets d’amers ponctuels 3D servent à générer différentes hypothèses de pose, retenant au final celle qui permet de prédire correctement le plus grand nombre de projections d’amers dans l’image courante. La pose ainsi calculée doit alors être intégrée au FKE afin d’en faire la nouvelle estimation courante de la pose. Ceci est réalisé simplement en remplaçant la dernière estimation de pose par celle qui vient d’être calculée. Étant donné que la nouvelle pose a été obtenue en dehors du FKE, il faut veiller à également mettre à jour les termes de corrélation de la matrice de covariance liant amers de la carte et nouvelle pose : seules les corrélations avec les amers ayant permis de calculer cette pose (i.e., ceux utilisés dans l’algorithme des “trois points”) ne sont pas mises à 0.

Mis à part sa rapidité (i.e., les traitements sont réalisables en temps réel à 30Hz) et son caractère incré-

mentiel, la solution proposée dans [Williams et al., 2007a] présente l'avantage majeur de ne pas reposer sur l'estimation courante de la pose pour la détection de fermeture de boucle. Nous avons déjà observé dans l'introduction de ce mémoire que cette estimation dérive de plus en plus au cours du temps, alors que le robot se déplace dans des parties encore inexplorées de l'environnement. De plus, en cas de kidnapping, cette estimation devient complètement incohérente. En conséquence, il peut être dangereux de prendre en compte cette information comme point de départ de la détection de fermeture de boucle. Ainsi, la méthode considérée ici peut être aussi bien utilisée pour la détection de fermeture de boucle ([Williams et al., 2008]), que pour récupérer suite à un kidnapping ([Williams et al., 2007b], [Williams et al., 2007a]). Par ailleurs, comme mentionné dans l'état de l'art de la partie I, cette solution tire sa rapidité d'une méthode simple pour la reconnaissance des amers (i.e., les Randomized Tree, [Lepetit and Fua, 2006]) : comme indiqué dans [Williams et al., 2008], cette méthode conduit à de nombreux faux positifs, ceux-ci étant par la suite écartés grâce à l'algorithme des "trois points". Il serait par conséquent intéressant d'observer les résultats obtenus dans des environnements présentant un aliasing perceptuel très important, comme ceux considérés dans les expériences rapportées dans ce mémoire.

### 9.3 Méthodes à base de filtrage particulaire

Dans le cadre du FKE, la gestion d'hypothèses multiples n'est pas possible. Par conséquent, une grande attention doit être portée à la procédure d'association de données, et ce particulièrement dans les situations de fermeture de boucle, étant donné que toutes les décisions prises lors de cette étape sont irrévocables et conditionnent les estimations futures. Les solutions proposées dans les approches évoquées dans la section 9.2 de cet état de l'art montrent à quel point la mise en oeuvre d'une procédure pertinente pour l'association de données dans le cadre du FKE est complexe. Le filtre particulaire Rao-Blackwellisé (RBpf, [Doucet et al., 2000]) offre la possibilité de relâcher cette contrainte, en permettant notamment la gestion des hypothèses multiples. Grâce au RBpf, la prise en compte des événements de fermeture de boucle est réalisable par nature pour les algorithmes de SLAM métrique, comme par exemple dans le cadre du *FastSLAM* [Montemerlo et al., 2003].

Dans le RBpf, l'état (i.e., la position et la carte) est "Rao-Blackwellisé" : les estimations de la position et de la carte sont découplées, rendant l'estimation de la carte conditionnellement dépendante de l'estimation de la position. La distribution de probabilité de la position est approximée grâce à un ensemble d'échantillons pondérés appelés *particules*. Pour chaque particule, une carte de l'environnement est maintenue, et les positions des amers peuvent être estimées de manière indépendante. Par exemple, pour le *FastSLAM*, chaque amer possède son propre FKE de dimension réduite. Le principal avantage du RBpf émane de la représentation multimodale de la distribution de probabilité de la position : chaque échantillon correspond à une hypothèse à part entière, représentant ainsi l'historique complet d'une trajectoire ainsi que la carte associée. Une particule reflète donc également l'ensemble des décisions d'association de données prises le long

d'une hypothèse de trajectoire. En choisissant d'utiliser le RBpf pour le SLAM, il est possible de prendre en compte explicitement les situations ambiguës où plusieurs hypothèses sont plausibles en échantillonnant l'espace des associations de données possibles ainsi que les hypothèses de trajectoires correspondantes [Montemerlo and Thrun, 2003].

Le cadre du RBpf a récemment été adapté avec succès à la vision monoculaire ([Eade and Drummond, 2006], [Karlsson et al., 2005], [Pupilli and Calway, 2006]) mais également à la stéréo-vision ([Barfoot, 2005], [Elinas et al., 2006], [Sim et al., 2005]). Cependant, malgré les avantages énumérés plus haut, les techniques de filtrage particulaire présentent tout de même certaines limitations. Notamment, il a été montré [Bailey et al., 2006] que ce genre d'approche souffre d'un problème de dégénérescence lors du ré-échantillonnage des particules, ce qui est très gênant lors des fermetures de boucle puisque qu'une partie des trajectoires passées et les cartes associées sont totalement oubliées. Cet "épuisement" pourrait être théoriquement évité en employant un ensemble de particules dont la cardinalité augmente de manière exponentielle avec la longueur de la trajectoire, ce qui est difficilement envisageable dans des environnements réalistes de taille raisonnable. A la place, des implémentations pratiques du RBpf comme celle présenté dans [Stachniss et al., 2004] reposent sur un ensemble de particules de taille fixe, mais emploient des politiques de ré-échantillonnage complexes qui exercent un compromis entre diversité des particules (i.e., afin d'éviter le problème d'épuisement), et distribution de probabilité a posteriori viable (i.e., lorsqu'une fermeture de boucle est réalisée, les échantillons cohérents avec cet évènement doivent recevoir une probabilité élevée).

## 9.4 Méthodes indépendantes de toute solution de SLAM

Les solutions exposées jusque-là sont toutes fortement liées à un cadre d'estimation pour le SLAM (i.e., FKE ou RBpf). Par conséquent, elles ne possèdent pas de modèle interne ni de représentation de l'environnement qui soient développés spécialement pour la tâche de détection de fermeture de boucle : elles reposent complètement sur une information qui est contrainte en qualité et en quantité par l'algorithme de SLAM. Une approche alternative qui découple clairement l'estimation de la position et de la carte de la détection de fermeture de boucle a été développée par les auteurs de [Ho and Newman, 2007] et de [Newman et al., 2006]. La méthode de détection de fermeture de boucle est basée sur la vision et repose sur la mise en oeuvre d'une matrice de similarité pour gérer les hypothèses et pour trouver les cycles dans la trajectoire d'un robot mobile (voir section 1.3.2 du chapitre 1 de la partie I). L'estimation de l'état est réalisée grâce à un algorithme de SLAM à états retardés basé sur le FKE et fonctionnant à partir de mesures renvoyées par un télémètre laser : l'historique complet de la trajectoire du robot est maintenu dans un vecteur d'état comme une séquence de positions estimées, avec les observations considérées comme des transformations relatives entre ces positions. Lorsqu'une fermeture de boucle a été détectée, les positions dans le vecteur d'état qui correspondent aux vues mises en correspondance grâce à la matrice de similarité sont alignées par le biais d'une procédure itérative d'appariements de scans laser. Ainsi, la totalité de la trajectoire estimée est mise à

jour en fonction de la fermeture de boucle, aboutissant en une estimation plus cohérente de la position et de l'orientation de du robot lorsqu'un cycle est achevé dans sa trajectoire. Cependant, il est regrettable que le processus sous-jacent pour l'estimation ne repose pas lui aussi sur la vision (on peut notamment mentionner les travaux de [Eustice et al., 2004] sur le SLAM à états retardés basés sur le FKE et la vision).

En abordant le problème de la détection de fermeture de boucle dans le cadre d'une méthode de filtrage en particulier (i.e., le FKE ou le RBpf), on se retrouve confronté à plusieurs limitations. Cela implique d'une part le développement d'heuristiques complexes pour l'association de données, utilisant une information qui est optimisée pour le SLAM et non pour la détection de fermeture de boucle. Cela requiert d'autre part la prise en compte des spécificités de la méthode de filtrage du SLAM (comme l'épuisement des particules par exemple). A l'inverse, en traitant le problème de la détection de fermeture de boucle dans un cadre dédié, il est possible de sélectionner à partir des différents capteurs l'information pertinente pour la tâche considérée, offrant ainsi l'opportunité de construire une représentation de l'environnement qui soit cohérente et adaptée. De plus, cela permet d'associer la solution ainsi obtenue avec tout type d'algorithme de SLAM, même ceux qui ne sont pas basés sur la vision.

## 9.5 Conclusion

D'après les solutions mentionnées dans cet état de l'art, il semble que l'on puisse distinguer les approches qui sont directement liées à un processus d'estimation pour le SLAM de celles qui sont indépendantes et qui peuvent être associées à différents formalismes pour le SLAM. Dans le premier cas, les méthodes mises en oeuvre pour la détection de fermeture de boucle sont généralement des heuristiques, qui peuvent dans certains cas être assez complexes, essayant de tirer profit de l'information contenue dans la carte et de celle renvoyée par les capteurs pour reconnaître les lieux déjà cartographiés. On peut notamment remarquer dans ce genre d'approche que l'emploi de la vision semble être plus à même de fournir une information consistante pour résoudre ce problème. Dans le second cas, le modèle de l'environnement est optimisé pour la tâche de reconnaissance des lieux passés, avec un cadre clairement établi pour aborder la problématique correspondante : les taux de succès s'en retrouvent dès lors grandement améliorés lorsque comparés aux heuristiques des approches "intégrées". En particulier, ce sont les performances dans des environnements présentant un aliasing perceptuel important qui sont principalement augmentées.

# Chapitre 10

## Recalage de position dans le SLAM métrique

Dans cette application, notre algorithme Bayésien de détection de fermeture de boucle (*BayesianLCD*) est utilisé en conjonction avec un algorithme de SLAM métrique visuel : MonoSLAM [Davison et al., 2004]. Le but des travaux décrits ici est de recalibrer la position de la caméra dans MonoSLAM suite à un kidnapping par une détection de fermeture de boucle.

### 10.1 Enjeux et difficultés

Plusieurs difficultés entrent en jeu dans ce type d'expérience. Pour commencer, il faut pouvoir détecter le kidnapping, afin d'arrêter temporairement le processus d'estimation de l'algorithme de SLAM métrique. En effet, la méthode de filtrage mise en oeuvre dans MonoSLAM ne permet pas de suivre d'hypothèses multiples au cours du temps : il s'agit d'une méthode de suivi de position. Ainsi, continuer le processus d'estimation sur la base des images acquises depuis la position suivant le kidnapping reviendrait à considérer que la caméra se situe dans un voisinage très proche de sa position avant le kidnapping. Dans cette situation, l'association de données conduirait à l'inférence d'une position et d'une orientation invalides.

La deuxième difficulté apparaît ensuite : après le kidnapping, il faut être capable de déterminer si la caméra se trouve dans une zone déjà cartographiée de l'environnement. C'est pour cette tâche que nous employons notre méthode de détection de fermeture de boucle. Pour cette étape, il est indispensable de pouvoir se fier sans hésitations aux informations apportées par l'algorithme de détection de fermeture de boucle. Dit autrement, cet algorithme ne doit pas proclamer être revenu en un lieu déjà visité si ce n'est pas le cas (i.e., il ne doit pas y avoir de faux positifs). Nous avons déjà insisté sur ce point dans les parties précédentes, et nous verrons dans cette partie encore une fois à quel point c'est crucial : dans une application au SLAM métrique, chercher à positionner géométriquement la caméra à une position erronée conduirait à

une dérive fatale de tout le processus d'estimation.

Enfin, la dernière difficulté majeure dans l'expérience proposée ici vient de l'association de données une fois que la localisation qualitative de la caméra a été retrouvée par l'algorithme de détection de fermeture de boucle : à ce stade, nous savons que la caméra est retournée dans une zone déjà cartographiée de l'environnement, cette zone est identifiée de manière symbolique (i.e., elle correspond à un lieu dans le modèle de l'environnement correspondant), et il faut maintenant calculer une position et une orientation viables pour la caméra. Pour cela, il faut inférer une localisation géométrique précise. Nous choisissons ici d'utiliser l'information retournée par l'algorithme de géométrie multi-vues (voir chapitre 2 de la partie I, section 2.3.3) comme point de départ pour l'inférence. A partir de là, nous employons une méthode d'association de données avancée pour raffiner l'estimation.

## 10.2 Aperçu de la solution mise en oeuvre

Alors que MonoSLAM maintient une estimation 3D de la position et de l'orientation de la caméra au fur et à mesure que les images sont acquises à 30Hz, BayesianLCD traque les fermetures de boucle en analysant une image sur 30 afin de fonctionner à une fréquence de 1Hz. Lorsque BayesianLCD détecte une fermeture de boucle, il communique avec MonoSLAM pour lui transmettre une nouvelle estimation de la *pose* (i.e., position et orientation) de la caméra : il s'agit de la pose corrigée d'après les informations de fermeture de boucle. Cette estimation est alors intégrée dans le processus d'inférence de MonoSLAM comme l'information provenant d'un capteur capable de réaliser des *observations directes* de la pose (à la manière d'un GPS par exemple). Une fois cette étape d'intégration achevée, une version modifiée et basée vision du JCT (Joint Compatibility Test, [Neira and Tardós, 2001]) est employée pour réaliser une association de donnée correcte et cohérente sur la base de la pose corrigée. Un diagramme détaillé du processus général conduisant à cette estimation est donné dans la figure 10.1.

## 10.3 Fonctionnement général de MonoSLAM

MonoSLAM est basé sur le filtre de Kalman étendu (FKE) pour réaliser l'estimation du vecteur d'état  $\mathbf{x}$  qui contient (voir [Davison, 2003] et [Davison et al., 2004] pour plus de détails) :

- le vecteur 3D de position ( $\mathbf{r}^W$ ) et le quaternion d'orientation ( $\mathbf{q}^{WR}$ ) de la caméra, la combinaison des deux formant la pose de la caméra
- les vitesses de translation ( $\mathbf{v}^W$ ) et de rotation ( $\omega^W$ ) de la caméra
- les positions 3D ( $\mathbf{y}_l$ ) des amers de la carte

La figure 10.2 donne une représentation des repères utilisés pour décrire les coordonnées du vecteur d'état. On y voit que les coordonnées des amers peuvent être exprimées dans deux référentiels distincts : le référentiel absolu  $W$  et le référentiel  $R$  relatif à la caméra.

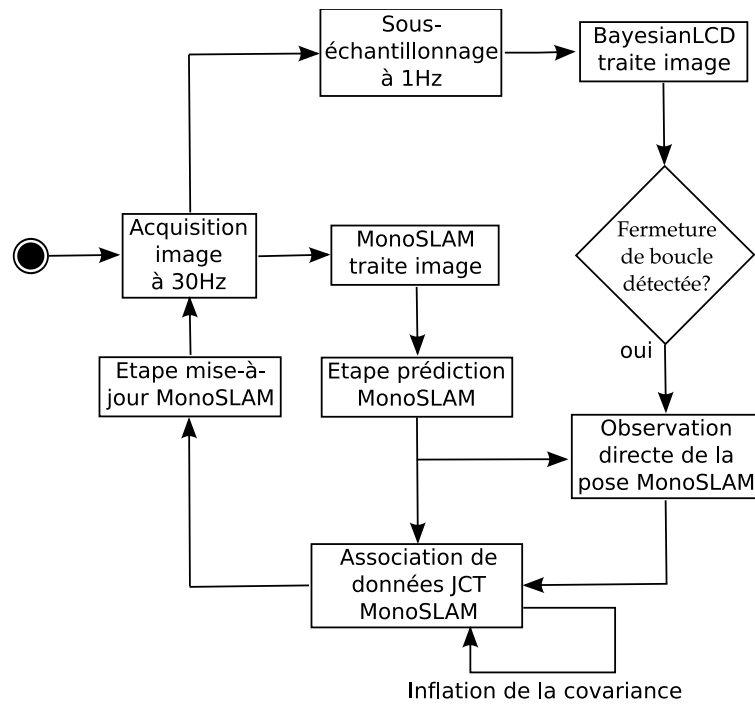


FIG. 10.1: Diagramme du processus global du recalage de la position dans l'algorithme de SLAM métrique (voir le texte pour les détails).

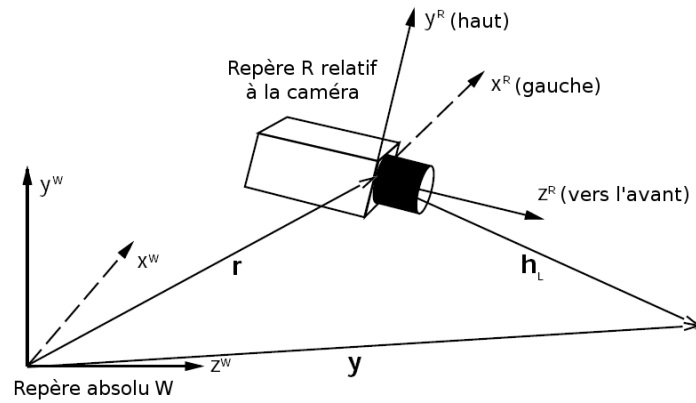


FIG. 10.2: Référentiels et systèmes de coordonnées. Source : [Davison et al., 2004].

D'après les notations employées dans [Davison et al., 2004], la valeur estimée d'une quantité a son symbole couvert par un chapeau  $\hat{\cdot}$ . Ainsi, le vecteur d'état estimé est noté  $\hat{x}$  et peut être écrit comme suit :

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{r}}^W \\ \hat{\mathbf{q}}^{WR} \\ \hat{\mathbf{v}}^W \\ \hat{\omega}^W \\ \hat{\mathbf{y}}_1 \\ \vdots \\ \hat{\mathbf{y}}_p \end{pmatrix} \quad (10.1)$$

Le processus d'estimation de MonoSLAM suit la procédure classique prédiction – mesure – mise-à-jour du FKE. Tout d'abord, MonoSLAM prédit la pose et les vitesses de la caméra à l'instant  $t$ , sur la base des dernières estimations de ces quantités et en se fiant à un modèle d'évolution temporelle :

$$\mathbf{f}_v = \begin{pmatrix} \mathbf{r}_{new}^W \\ \mathbf{q}_{new}^{WR} \\ \mathbf{v}_{new}^W \\ \omega_{new}^W \end{pmatrix} = \begin{pmatrix} \mathbf{r}^W + (\mathbf{v}^W + \mathbf{V}^W) \Delta t \\ \mathbf{q}^{WR} \times \mathbf{q}((\omega^W + \Omega^W) \Delta t) \\ \mathbf{v}^W + \mathbf{V}^W \\ \omega^W + \Omega^W \end{pmatrix} \quad (10.2)$$

Dans le modèle d'évolution temporelle de MonoSLAM, on considère qu'entre deux instants séparés par un temps  $\Delta t$ , les vitesses de translation et de rotation de la caméra sont respectivement perturbées par des bruits Gaussiens  $\mathbf{V}^W$  et  $\Omega^W$  de moyennes nulles et dont les écarts types reflètent un mouvement sans accélérations brusques. On considère également que les amers de la carte restent immobiles (c'est pourquoi ils n'apparaissent pas dans l'équation 10.2, leur état restant inchangé). Par ailleurs, la notation “ $\mathbf{q}((\omega^W + \Omega^W) \Delta t)$ ” dénote le quaternion obtenu à partir du vecteur de rotation angle-axe défini par “ $(\omega^W + \Omega^W) \Delta t$ ” (cf. [Davison, 2003] pour plus de détails).

Après l'étape de prédiction, les amers 3D de la carte qui sont visibles depuis la pose prédite sont projetés dans le plan image de la caméra, afin de pouvoir être comparés avec les primitives visuelles qui ont été extraites dans l'image courante. Pour cela, on commence par transformer les coordonnées absolues de ces amers en coordonnées relatives à la pose prédite pour la caméra. D'après les notations de la figure 10.2, cela se traduit ainsi :

$$\mathbf{h}_L^R = \mathbf{R}^{RW} (\mathbf{y}_l^W - \mathbf{r}^W) \quad (10.3)$$

avec  $\mathbf{R}^{RW}$  la matrice de rotation exprimant le changement d'orientation entre le référentiel relatif  $R$  et le référentiel absolu  $W$ . Celle-ci peut être obtenue simplement à partir du quaternion  $\mathbf{q}^{WR}$  d'orientation du repère de la caméra dans le référentiel absolu.

On peut ensuite projeter ces coordonnées 3D dans le plan image de la caméra :



$$\mathbf{h}_l = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u_0 - f k_u \frac{h_{Lx}^R}{h_{Lz}^R} \\ v_0 - f k_v \frac{h_{Ly}^R}{h_{Lz}^R} \end{pmatrix} \quad (10.4)$$

où  $f k_u$  (respectivement  $f k_v$ ) représente la distance focale exprimée en unité horizontale (respectivement verticale) de pixels, alors que  $u_0$  et  $v_0$  correspondent aux coordonnées du point principal de la caméra. Dans [Davison et al., 2004], le lecteur intéressé trouvera la dernière étape de cette projection, visant à annuler la distorsion radiale due à l'objectif grand-angle utilisé dans les expériences.

Lorsque la projection dans l'image courante des amers visibles depuis la pose prédite est terminée, l'association de données opère : les amers projetés doivent être correctement appariés avec les primitives mesurées dans l'image. Une fois que cette procédure est achevée, l'innovation (i.e., les distances séparant les projections d'amers des primitives auxquelles ils ont été associés) est propagée sur l'ensemble des quantités du vecteur d'état par le biais du *gain de Kalman* (voir notamment [Dissanayake et al., 2001] pour plus de détails sur la manière dont ce gain est calculé). C'est cette innovation qui sert à corriger à la fois les paramètres régissant la localisation de la caméra et les positions des amers de la carte. Le gain de Kalman sert quant à lui à diffuser l'innovation sur les différentes quantités du vecteur d'état, notamment en fonction de l'incertitude qui leur est associée.

### 10.3.1 Association de données dans MonoSLAM

La méthode originellement mise en oeuvre dans MonoSLAM pour l'association de données repose sur la distance de Mahalanobis pour mettre simplement en correspondance chaque amer projeté avec son plus proche voisin parmi les primitives de l'image (cf. figure 10.3). Ces primitives sont sélectionnées dans des régions elliptiques centrées sur les projections des amers et dont la taille dépend de l'incertitude. Plus précisément, ces ellipses correspondent aux projections dans l'image des ellipses d'incertitude 3D associées aux amers de la carte. Comme nous allons le montrer dans le chapitre relatif aux résultats expérimentaux de cette partie, cette technique simple pour l'association de données n'est pas suffisante pour détecter les fermetures de boucle.

En conséquence, nous avons modifié cette procédure afin de prendre en compte des contraintes de géométrie locale pour sélectionner un ensemble de paires {amer de la carte – primitive de l'image} qui sont conjointement compatibles, comme décrit dans [Neira and Tardós, 2001] et [Neira et al., 2003]. Pour cela, on commence par chercher plusieurs primitives visuelles dans l'image qui ressemblent à un amer donné, alors que dans MonoSLAM, cette recherche n'aboutit qu'à une seule hypothèse d'association. Il faut donc chercher, à l'intérieur des régions elliptiques décrites plus haut, plusieurs primitives qui ressemblent à l'amer centré sur la zone de recherche. On obtient ainsi, pour chacun des amers visibles depuis la pose courante, un ensemble de primitives visuelles plausibles. On compare ensuite les distances relatives entre projections d'amers et entre primitives visuelles correspondantes. Cela permet de filtrer les hypothèses d'association à

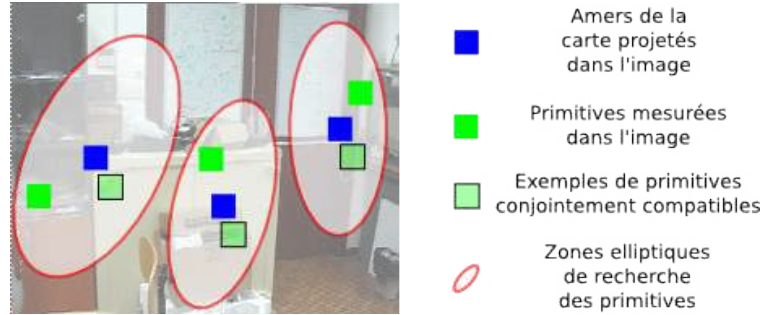


FIG. 10.3: Méthode d'association de données mise en oeuvre originellement dans [Davison et al., 2004]. Les amers visibles depuis la pose prédite sont projetés dans l'image courante (carrés bleus). L'incertitude associée à ces amers et à la pose prédite sert à définir les zones de recherche pour l'association de données (ellipses rouges). A l'intérieur de chaque zone de recherche, on ne retient que la primitive extraite dans l'image (carré vert) qui ressemble le plus à l'amer projeté au centre de l'ellipse : cette primitive est alors considérée comme la mesure de l'amer correspondant dans l'image. Comme on peut le remarquer, cette procédure peut conduire à une association de données conjointement incompatible : dans chacune des ellipses, les positions des primitives relativement aux amers sont toutes différentes. Avec une association de données conjointement compatible, chaque primitive se situerait au même endroit que les autres dans les zones de recherche (carrés vert-pâle entouré de noir).

un premier niveau, en écartant les hypothèses pour lesquelles ces distances ne sont pas comparables. Cette première étape de filtrage, illustrée dans la figure 10.4, est suggérée dans [Neira et al., 2003] afin de réduire les traitements suivants. Finalement, le reste des hypothèses est soumis au JCT ([Neira and Tardós, 2001]), celui-ci permettant de s'assurer que l'ensemble des primitives choisies pour l'appariement avec les amers projetés s'accordent toutes sur l'innovation à appliquer : on empêche de cette manière d'avoir une primitive donnant une correction dans une direction opposée à la direction proposée par le reste des primitives par exemple.

### 10.3.2 Joint Compatibility Test

Le JCT ([Neira and Tardós, 2001]) permet de tester la cohérence de l'association de données pour un ensemble de paires {amer de la carte – primitive de l'image}, par l'imposition d'un seuil sur l'innovation conjointe calculée à partir de cet ensemble. Comme déjà expliqué précédemment, l'innovation est donnée par la distance cartésienne séparant la projection  $\hat{\mathbf{h}}_l$  d'un amer dans l'image, de la mesure  $\hat{\mathbf{z}}_m$  de la primitive correspondant à cet amer :

$$\nu_{ml} = \hat{\mathbf{z}}_m - \hat{\mathbf{h}}_l \quad (10.5)$$

$\hat{\mathbf{z}}_m$  est la mesure bruitée dans l'image de la primitive  $\mathbf{z}_m$  : dans le cadre du FKE employé dans Mono-SLAM, la position réelle de la primitive est perturbée par un bruit blanc Gaussien de moyenne nulle (cf. [Davison et al., 2004]). On peut alors concaténer les innovations respectives de chacune des paires cor-

respondant à une association de données dans un vecteur colonne, pour former le vecteur de l'innovation conjointe :

$$\mathcal{V} = \begin{pmatrix} \nu_{m_1 l_1} \\ \vdots \\ \nu_{m_r l_r} \end{pmatrix}$$

Pour évaluer la compatibilité conjointe de l'ensemble de paires considéré, on calcule une distance de Mahalanobis sur l'innovation conjointe, et on impose que celle-ci soit inférieur à un certain seuil :

$$D_{\mathcal{V}}^2 = \mathcal{V}^T \mathbf{C}_{\mathcal{V}}^{-1} \mathcal{V} < \chi_{d,\alpha}^2 \quad (10.6)$$

où  $\mathbf{C}_{\mathcal{V}}^{-1}$  correspond à la covariance de l'innovation (les détails correspondant sont donnés dans [Neira and Tardós, 2001]). Si l'équation 10.6 est vérifiée, le JCT est validé et l'ensemble de paires est considéré comme conjointement compatible : l'association de données correspondante est entérinée.

Le seuil  $\chi_{d,\alpha}^2$  est obtenu à partir d'une distribution de probabilité *chi-carré*, et il est paramétré par le nombre de degrés de liberté  $d = \dim(\mathcal{V})$  et par le niveau de confiance désiré  $\alpha$  (choisi généralement entre 0.95 et 0.99). L'idée est de considérer l'association de données représentée par  $\mathcal{V}$  comme une "hypothèse nulle" que l'on admet comme vraie tant qu'aucune preuve ne supporte une hypothèse alternative avec un niveau de confiance égal à  $\alpha$ . Il est important de noter ici que le niveau de confiance porte sur le rejet de l'hypothèse nulle : plus on est tolérant sur ce niveau de confiance, plus on admet l'éventualité d'une hypothèse alternative sans forcément requérir de preuves fortes à son égard. Ainsi, plus la valeur de  $\alpha$  est faible, moins le support requis pour écarter l'hypothèse nulle et admettre l'hypothèse alternative doit être important. Par conséquent, la valeur du seuil imposé pour valider le JCT diminue quand  $\alpha$  diminue, ce qui rend la validation de l'association plus difficile (mais cela apporte également plus de certitude dans le résultat).

Afin de faciliter la mise en oeuvre du JCT pour l'association de données, les auteurs de [Neira and Tardós, 2001] proposent une méthode incrémentielle pour son calcul : cela permet d'évaluer la viabilité de l'ajout d'une paire {amer de la carte – primitive de l'image} à un ensemble déjà conjointement compatible.

D'une manière générale, l'emploi du JCT pour l'association de données permet d'améliorer grandement la qualité de l'estimation, limitant les dérives dues au bruit local dans l'image. Notamment, cette amélioration est remarquable dans les résultats présentés dans [Clemente et al., 2007], où le JCT a été employé dans la version *inverse depth*<sup>1</sup> de MonoSLAM.

<sup>1</sup>Originellement, dans MonoSLAM, chaque amer est localisé par ses coordonnées cartésiennes exprimées dans un repère absolu. Dans la version "inverse depth", chaque amer est localisé par la position de la caméra lorsqu'il a été observé la première fois, et par les coordonnées relatives (azimut, élévation, et inverse de la profondeur) de cet amer par rapport au point de vue de la caméra. Le choix de l'inverse de la profondeur présente des caractéristiques intéressantes pour la modélisation du bruit associé sous la forme d'une Gaussienne. La version "inverse depth" de MonoSLAM est détaillée dans [Civera et al., 2006].

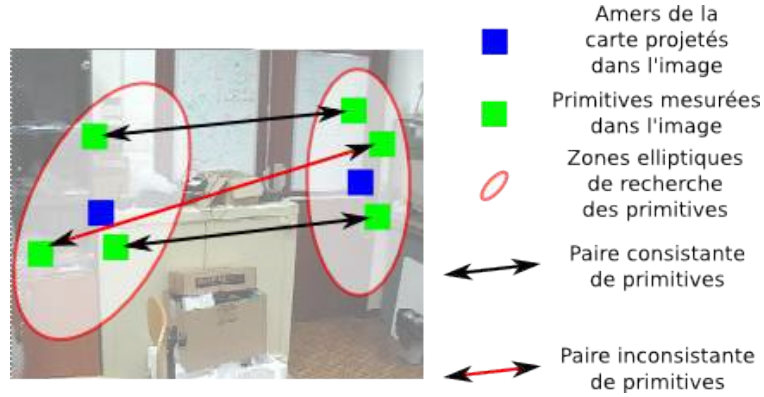


FIG. 10.4: Association de données multi-hypothèses : uniquement les primitives visuelles de l'image courante qui satisfont les contraintes de géométrie locale (i.e., les distances relatives ici) sont ensuite soumises au JCT.

## 10.4 Détection de fermeture de boucle

En parallèle de l'estimation récursive réalisée dans MonoSLAM, BayesianLCD traite une partie des images pour détecter les fermetures de boucles, comme cela a été décrit dans le chapitre 2 de la partie I. Nous avons ainsi vu que pour être acceptée, une hypothèse de fermeture de boucle devait vérifier la contrainte de géométrie épipolaire, testée dans BayesianLCD grâce à un algorithme de géométrie multi-vues. En cas de succès, ce dernier retourne la transformation (i.e., rotation et translation) permettant de passer du point de vue du lieu reconnu vers le point de vue de l'image courante. Par conséquent, après une détection de fermeture de boucle, on peut aisément obtenir une nouvelle estimation de la pose actuelle en composant cette transformation relative avec la pose du lieu de fermeture de boucle (voir figure 10.5). La pose de ce lieu provient de l'estimation réalisée par MonoSLAM au moment du passage antérieur de la caméra à cet endroit.

### 10.4.1 Modèle d'observation directe de la pose

La pose corrigée doit maintenant être intégrée dans le processus d'estimation de MonoSLAM. Pour cela, nous avons développé un modèle d'observation directe de la pose : on considère la pose corrigée renvoyée par BayesianLCD comme une mesure obtenue à partir d'un capteur capable de fournir une observation directe de la pose de la caméra. Un tel modèle d'observation peut être obtenu en écrivant l'équation qui lie les quantités estimées (i.e., la totalité du vecteur d'état  $\mathbf{x}$ ) aux quantités observées (i.e., la position et l'orientation de la caméra) :

$$\mathbf{h}_{\text{od}} = \begin{pmatrix} \mathbf{r}^W \\ \mathbf{q}^{WR} \end{pmatrix} \quad (10.7)$$

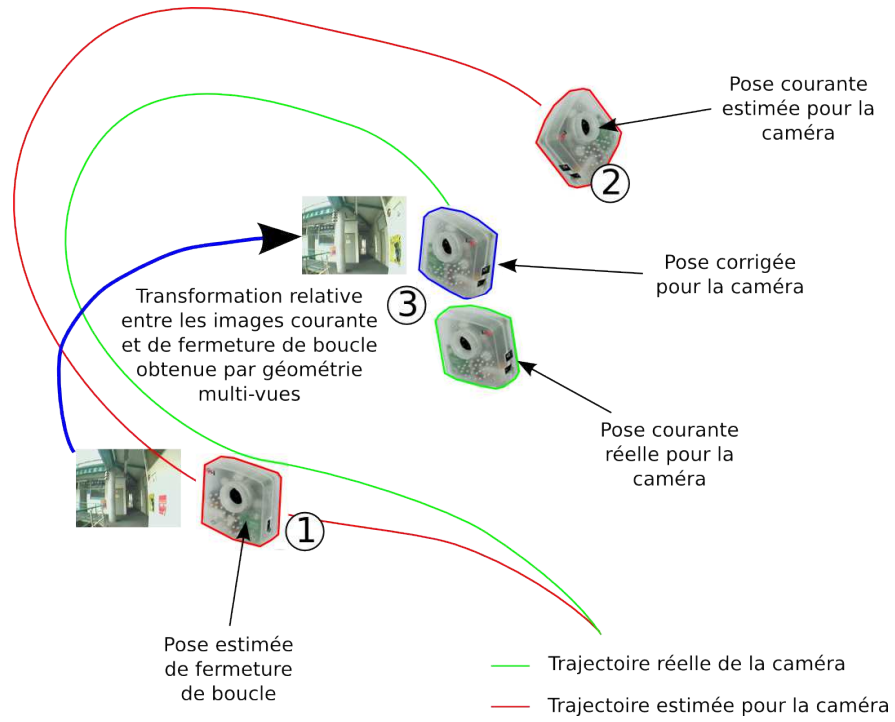


FIG. 10.5: Correction de la pose dans MonoSLAM après une détection de fermeture de boucle. Une fermeture de boucle a été détectée entre les images provenant d'une pose estimée précédemment (1) et de la pose actuellement estimée (2). On peut composer la pose de la caméra estimée précédemment avec la transformation relative obtenue à partir de ces images pour obtenir la pose corrigée pour la caméra (3) qui correspond à une meilleure estimation de la pose réelle de la caméra.

Ici,  $\mathbf{h}_{\text{od}}$  est simplement une version réduite du vecteur d'état  $\mathbf{x}$  qui contient uniquement la position  $\mathbf{r}^W$  et l'orientation  $\mathbf{q}^{WR}$  de la caméra relativement à un référentiel absolu fixe. D'après ce modèle d'observation, on peut mettre à jour la pose estimée pour la caméra dans le FKE de MonoSLAM d'après l'équation suivante :

$$\hat{\mathbf{x}}(\mathbf{k} + 1) = \hat{\mathbf{x}}(\mathbf{k}) + \mathbf{K}_{\text{od}}(\tilde{\mathbf{h}}_{\text{od}} - \hat{\mathbf{h}}_{\text{od}}(\mathbf{k})) \quad (10.8)$$

Dans l'équation 10.8,  $\tilde{\mathbf{h}}_{\text{od}}$  est la pose renvoyée par BayesianLCD, alors que  $\hat{\mathbf{h}}_{\text{od}}(\mathbf{k})$  est l'observation directe de la pose basée sur les quantités estimées à l'instant  $\mathbf{k}$  (cf. équation 10.7).  $\mathbf{K}_{\text{od}}$  est le gain de Kalman requis pour propager l'innovation obtenue à partir des quantités observées à l'ensemble du vecteur d'état. L'indice  $\mathbf{k}$  sert à distinguer l'état des quantités estimées avant et après la mise à jour. Le gain de Kalman  $\mathbf{K}_{\text{od}}$  peut être obtenu comme suit :

$$\mathbf{K}_{\text{od}} = \mathbf{P}\mathbf{H}_{\text{od}}^T (\mathbf{H}_{\text{od}}\mathbf{P}\mathbf{H}_{\text{od}}^T + \mathbf{R}_{\text{od}})^{-1} \quad (10.9)$$

où  $\mathbf{P}$  est la matrice de covariance associée au vecteur d'état, alors que  $\mathbf{H}_{\text{od}}$  et  $\mathbf{R}_{\text{od}}$  sont respectivement la matrice jacobienne et la matrice de bruit du modèle d'observation directe de la pose.

Afin de faire converger la pose estimée pour la caméra  $\hat{\mathbf{x}}(\mathbf{k} + 1)$  vers la pose corrigée  $\tilde{\mathbf{h}}_{\text{od}}$ , celle-ci est “mesurée” dans le modèle d'observation avec un bruit proche de zéro, simulant ainsi une mesure capteur avec un niveau de confiance tendant vers l'infini. Pour cela, la matrice  $\mathbf{R}_{\text{od}}$  est diagonale avec des éléments proches de zéro. Par ailleurs, la matrice jacobienne  $\mathbf{H}_{\text{od}}$  est formée par les dérivées partielles de  $\mathbf{h}_{\text{od}}$  par rapport au vecteur d'état :

$$\mathbf{H}_{\text{od}} = \frac{\partial \mathbf{h}_{\text{od}}}{\partial \mathbf{x}} \quad (10.10)$$

Ici,  $\mathbf{H}_{\text{od}}$  est ici simplement une matrice de sélection :

$$\mathbf{H}_{\text{od}} = \begin{pmatrix} \mathbf{Id}_{7 \times 7} & \mathbf{0}_{7 \times (3p+6)} \\ \mathbf{0}_{(3p+6) \times 7} & \mathbf{0}_{(3p+6) \times (3p+6)} \end{pmatrix}$$

où  $\mathbf{Id}_{N \times N}$  désigne la matrice identité de taille  $N \times N$ , alors que  $\mathbf{0}_{N \times N}$  désigne la matrice nulle de taille  $N \times N$ .

Les détails concernant la mise à jour de la matrice de covariance  $\mathbf{P}$  ne sont pas donnés ici. Encore une fois, le lecteur intéressé peut se référer à [Dissanayake et al., 2001] pour plus d'informations à ce sujet.

#### 10.4.2 Inflation de la covariance

A partir de l'équation 10.8, on obtient une nouvelle estimation pour le vecteur d'état, avec une position et une orientation corrigées. Il faut maintenant réaliser l'association de données dans le but d'apparier les primitives extraites dans l'image courante avec les amers de la carte qui sont visibles d'après le point de vue de la pose corrigée. Toutefois, il faut considérer deux points qui rendent la procédure d'association de données particulièrement difficile ici.

Tout d'abord, la transformation relative calculée par l'algorithme de géométrie multi-vues n'est qu'une approximation assez grossière du changement de point de vue existant en réalité. En effet, les algorithmes de géométrie multi-vues se basent généralement sur plusieurs images (i.e., plus de deux) afin d'obtenir une estimation précise de cette transformation (cf. par exemple [Eustice et al., 2004], [Hartley and Zisserman, 2004], [Horn, 1990] et [Nistér et al., 2006]). Par conséquent, les amers de la carte se retrouvent projetés ici à des positions éloignées des primitives correspondantes dans l'image courante.

Deuxièmement, la taille des régions elliptiques de recherche des primitives visuelles dépend en partie du bruit associé au modèle d'observation des amers mis en oeuvre dans MonoSLAM. Ce bruit a été pensé dans le cadre du suivi d'amers dans des images consécutives, avec des changements de faible amplitude entre les points de vue correspondants. Dans le cas d'une fermeture de boucle au contraire, ces variations sont beaucoup plus importantes.

Pour outrepasser ces difficultés, nous avons incorporé l'étape d'association de données dans une procédure récursive d'inflation de la covariance visant à régulièrement augmenter l'incertitude dans la matrice de covariance  $\mathbf{P}$  associée au vecteur d'état  $\mathbf{x}$ . En augmentant artificiellement le niveau d'incertitude sur la pose de la caméra, on augmente l'incertitude associée aux amers projetés dans l'image courante, et on obtient en conséquence des ellipses de recherche plus importantes dans l'image pour la mise en correspondance des amers avec les primitives visuelles (voir figure 10.6). Cela se traduit par un nombre plus important d'hypothèses plausibles pour l'association de données : plus l'incertitude est importante, plus le nombre d'hypothèses plausibles à vérifier est grand.

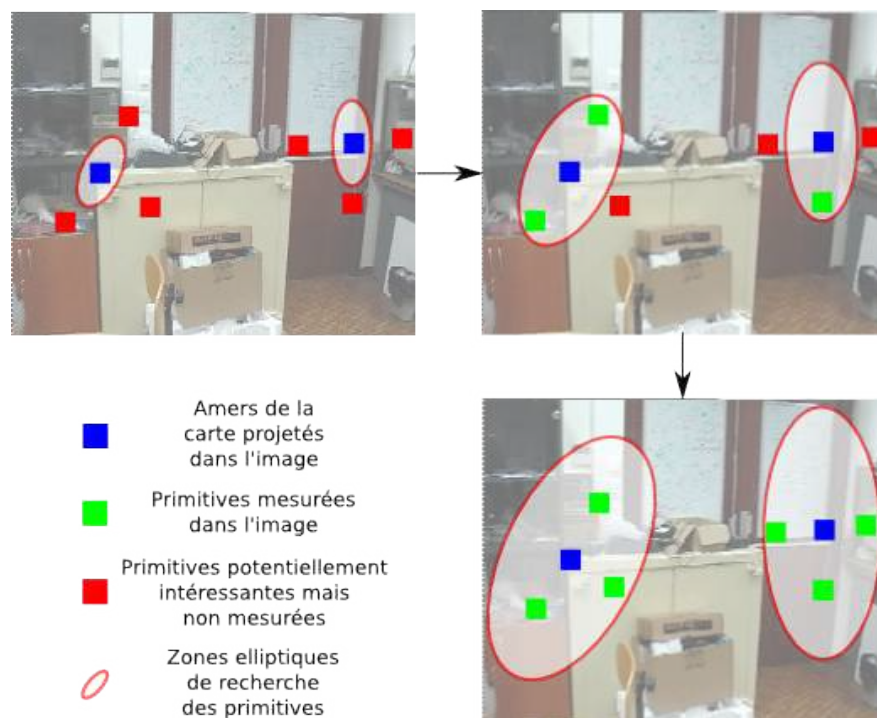


FIG. 10.6: Illustration de l'inflation de la covariance. En augmentant l'incertitude dans la matrice de covariance, on augmente la taille des ellipses de recherche dans l'image. Dans la figure, l'incertitude initiale empêche d'apparier correctement les amers projetés (image en haut à gauche) : en raison de la trop petite taille des zones de recherche, certaines primitives ne sont pas mesurées alors qu'elles sont potentiellement intéressantes pour l'appariement (carré rouges). En augmentant l'incertitude, on augmente la taille des zones de recherche : ces primitives sont alors mesurées et prises en compte pour l'appariement.

Cette procédure d'inflation de la covariance est répétée suite à une détection de fermeture de boucle jusqu'à ce qu'au moins un nombre  $a$  d'amers projetés dans l'image courante aient été mis en correspondance avec des primitives visuelles au sens du JCT, comme décrit précédemment. Le nombre  $a$  est fixé initialement à la moitié du nombre d'amers visibles et sélectionnés par MonoSLAM pour l'appariement (voir [Davison, 2003]). Cette valeur de départ peut ensuite être modifiée sur la base du niveau de qualité de l'association

de données : plus cette qualité est élevée, plus l'association de données est sûre. Ainsi, pour une qualité importante, on requiert un nombre plus faible d'appariements, alors que si cette qualité est peu élevée, on impose que le nombre d'appariements soit plus grand. Le niveau de qualité est évalué en fonction de la confiance désirée pour le JCT (i.e., le paramètre  $\alpha$  de la distribution  $\chi^2_{d,\alpha}$  dans l'équation 10.6) : on cherche la valeur la plus faible pour  $\alpha$  prise entre 0.95 et 0.99 pour laquelle l'équation 10.6 est validée. Plus cette valeur est basse, plus la qualité est élevée (i.e., il est important de rappeler ici que plus la valeur de  $\alpha$  est faible, plus les contraintes requises pour valider l'association de données sont importantes, cf. section 10.3.2).

L'inflation de la covariance de la pose de la caméra peut-être réalisée facilement en multipliant les éléments correspondants sur la diagonale de  $\mathbf{P}$ . Nous avons ici choisi un coefficient pour cette multiplication correspondant à une augmentation de 10% à chaque itération. Une procédure similaire d'inflation de la covariance est également utilisée dans [Williams et al., 2007b] pour récupérer suite à une perte du suivi de position de la caméra.

On peut immédiatement remarquer d'après le principe de correction de la pose de la caméra mis en oeuvre ici que celui-ci repose sur une confiance totale dans l'algorithme de détection de fermeture de boucle. En effet, lors d'une fermeture de boucle, la procédure d'inflation de la covariance augmente l'incertitude par paliers de 10% jusqu'à ce qu'une association de données cohérente soit trouvée. Cela laisse sous-entendre que l'on est sûr d'être revenu sur une position passée et que l'on cherche à déterminer précisément la nouvelle pose. Il est donc indispensable qu'aucun faux positifs (i.e., quand l'algorithme proclame être revenu sur une position passée si ce n'est pas le cas) ne soient détectés par BayesianLCD.

On peut également émettre une deuxième remarque au sujet du critère retenu pour entériner définitivement une pose corrigée suite à la procédure d'association de données : on attend qu'un certain nombre des amers visibles d'après la pose corrigée pour la caméra soient appariés avec des primitives dans l'image au sens du JCT. Le faible pouvoir descriptif et le manque de généralisation du descripteur utilisé pour caractériser les amers de la carte dans MonoSLAM empêche de chercher à apparier 100% de ces amers suite à une fermeture de boucle. Une faible variation dans la luminosité de l'environnement entre les deux passages suffit à faire faillir l'association de données. Cela renforce deux des choix effectués ici. Tout d'abord, en ce qui concerne le modèle de l'environnement dédié pour la tâche de détection de fermeture de boucle : en reposant sur un modèle optimisé, avec des primitives robustes, on obtient un bon taux de reconnaissance même en cas de changements infimes dans l'environnement. Deuxièmement, en ce qui concerne la procédure d'association de données employant le JCT : cette procédure prend en compte les informations de géométrie locale dans l'image pour palier les faiblesses des descripteurs visuels.



# Chapitre 11

## Résultats expérimentaux

Dans ce chapitre, nous présentons les résultats expérimentaux obtenus à partir d'une séquence vidéo acquise dans un environnement d'intérieur (voir figure 11.1 pour un aperçu de la pièce dans laquelle l'expérience a été réalisée). Pour réaliser cette vidéo, une caméra grand-angle ( $120^\circ$  d'angle de vue, réglage automatique de l'exposition, images de taille 320x240 pixels) a été déplacée à la main selon une trajectoire formant une boucle. Plusieurs aspects relatifs aux modifications apportées à MonoSLAM sont abordés dans ce chapitre. Nous démontrons dans un premier temps à quel point le JCT permet d'améliorer la procédure d'association de données et, par conséquent, le processus d'estimation de manière générale. Pour cela, nous comparons simplement la trajectoire de la caméra estimée sans et avec le JCT : dans le premier cas, la fermeture de boucle n'est pas détectée, alors qu'elle l'est dans le second. Nous démontrons ensuite la qualité de la solution décrite dans cette partie du mémoire en réalisant une expérience de kidnapping, afin de prouver qu'il est possible de rétablir une estimation correcte de la pose de la caméra dans ces conditions même dans le cadre d'une méthode de filtrage ne permettant pas de gérer plusieurs hypothèses simultanément.

Ce chapitre est donc découpé en sections dédiées aux différents aspects énoncés ci-dessus. Les résultats présentés ici ont fait l'objet de la soumission d'un article en cours de revue. Par ailleurs, une vidéo présentant l'expérience de kidnapping est disponible au téléchargement à l'adresse suivante : <http://animatlab.lip6.fr/AngeliVideosFr>.

### 11.1 Amélioration de la procédure d'association de données

Ce chapitre débute par l'expérience destinée à montrer l'intérêt d'utiliser le JCT pour réaliser l'association de données dans MonoSLAM. L'expérience consiste simplement à analyser une vidéo où la trajectoire de la caméra suit un parcours cyclique, afin de tester la détection de fermeture de boucle avec et sans le JCT. On peut voir d'après la figure 11.2 que lorsque l'association de données est effectuée selon la procédure standard du plus proche voisin (PPV), ainsi que défini par défaut dans MonoSLAM, la trajectoire



FIG. 11.1: Aperçu de la pièce dans laquelle l'expérience a eu lieu.

de la caméra est incohérente étant donné que le cycle n'est pas détecté et qu'en conséquence, la boucle n'est pas fermée. À l'inverse, en employant le JCT pour l'association de données, le cycle est correctement achevé : le système prédit correctement la position d'anciens amers de la carte lorsqu'ils deviennent visibles à nouveau, alors que la caméra revient dans une zone déjà cartographiée. Cela laisse donc sous-entendre que l'estimation réalisée grâce au JCT est plus cohérente. Il est important de noter ici que la fermeture de boucle a été réalisée sans reposer sur l'algorithme BayesianLCD : le cycle dans la trajectoire a été achevé seulement parce que les quantités estimées (i.e., la pose de la caméra et la carte) sont inférées avec plus de précision que dans le cas où le PPV est employé. En conséquence, les projections des amers de la carte enregistrés lors d'un premier passage de la caméra dans un lieu donné sont réalisées de manière cohérente lorsque la caméra retourne dans ce lieu : l'association de données avec les primitives actuellement perçues s'en retrouve alors facilitée.

## 11.2 Expérience de kidnapping

Toutefois, l'implémentation simple du JCT réalisée ici pour l'association de données ne peut pas être employée pour récupérer de situations plus complexes, comme c'est le cas dans le scénario du robot kidnappé. L'expérience correspondante consiste à déplacer arbitrairement le robot alors que celui-ci a débuté son processus de localisation et de cartographies simultanées. La difficulté vient du fait que lorsque le robot

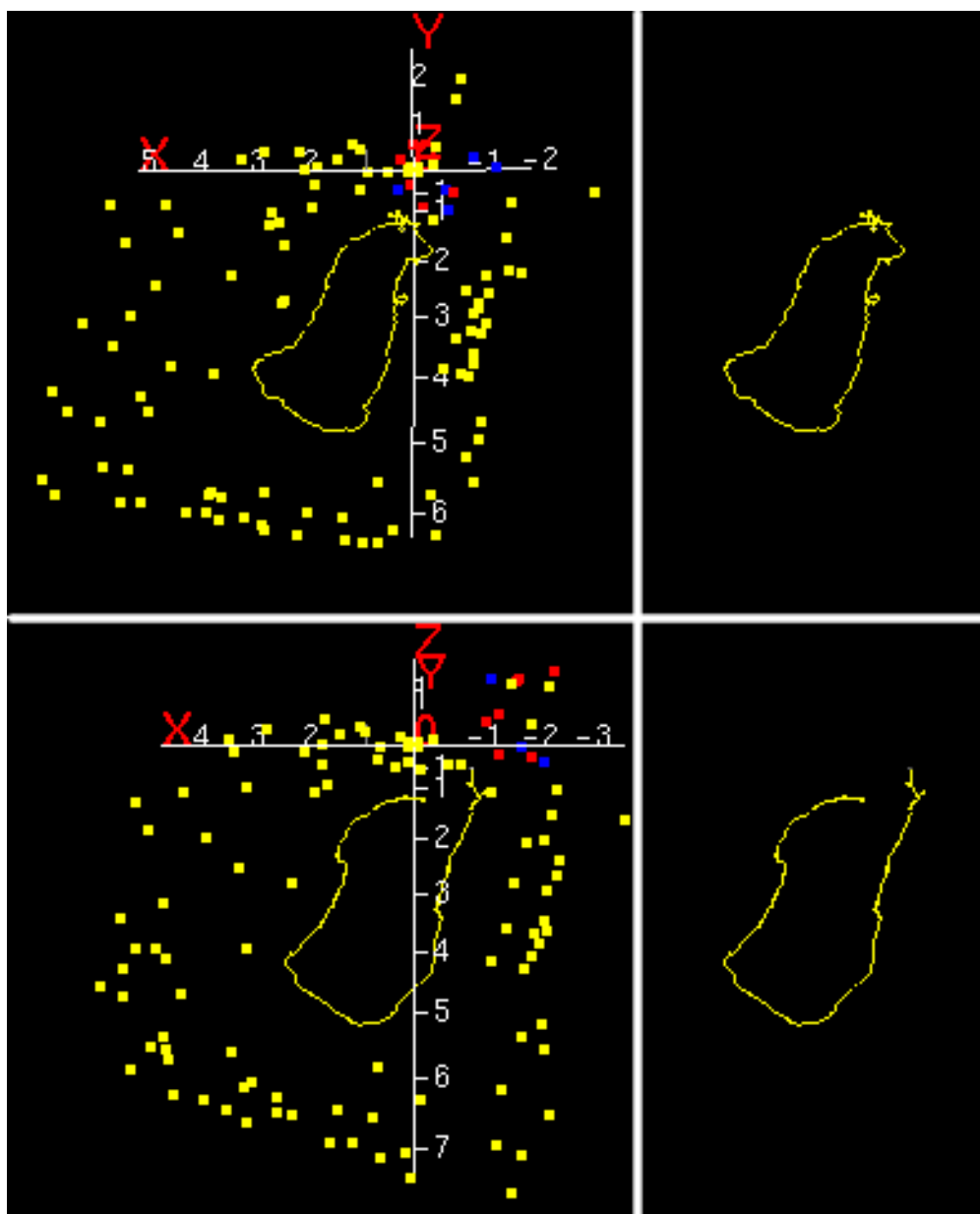


FIG. 11.2: Trajectoire de la caméra lorsque la procédure d'association de données est réalisée avec (haut) et sans (bas) le JCT : dans le premier cas, la fermeture de boucle est détectée correctement, même sans reposer sur l'algorithme BayesianLCD, alors que dans le second cas, la trajectoire estimée n'est pas cohérente. Sur chaque rangée, la partie gauche de la figure montre la trajectoire de la caméra et la carte correspondante, alors que sur la partie de droite, seulement la trajectoire est donnée.

est déplacé, il n'a aucune connaissances sur le trajet effectué lors de ce déplacement : une fois déposé dans l'environnement suite au kidnapping, il doit reconnaître le lieu dans lequel il se trouve si celui-ci a déjà été

cartographié, puis reprendre le processus de SLAM.

Ainsi, suite à un kidnapping, notre implémentation du JCT ne permettra pas de localiser correctement le robot. En effet, lors de l'association de données, les primitives extraites dans l'image courante ne sont comparées qu'aux amers actuellement visibles d'après la pose prédite pour la caméra, et non aux amers de toute la carte. Par conséquent, suite à un kidnapping, étant donné que la pose estimée pour la caméra n'est pas cohérente, l'association de données ne permettra peut-être pas de trouver les amers de la carte qui correspondent aux primitives extraites depuis la pose suivant le kidnapping (cf. figure 11.3).

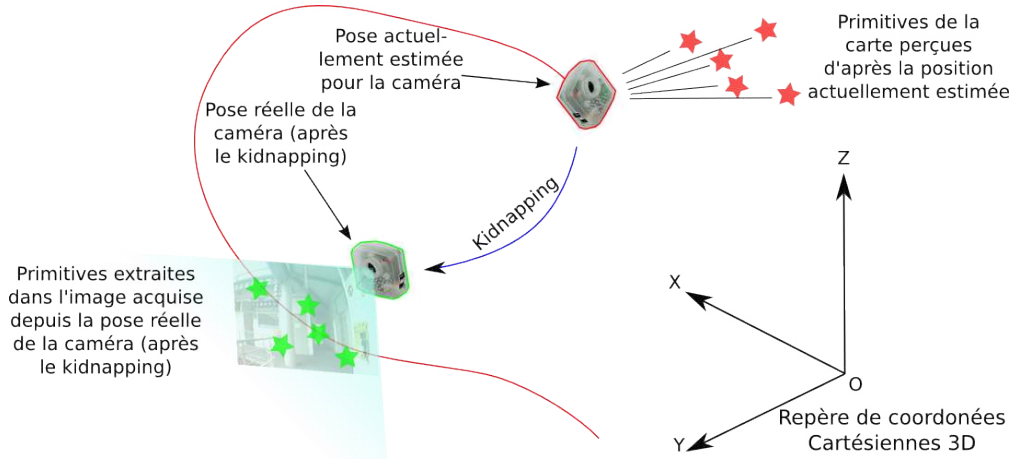


FIG. 11.3: Après un kidnapping, il est risqué de comparer les amers de la carte perçus depuis la pose estimée pour la caméra avec les primitives mesurées dans l'image courante acquise depuis la pose réelle (i.e., après le kidnapping) : en effet, la distance séparant les projections des amers dans l'image des primitives trouvées peut être importante.

D'après cette constatation, on déduit qu'il est indispensable de pouvoir suspendre le processus d'estimation du SLAM tant qu'une fermeture de boucle n'est pas détectée grâce à l'algorithme BayesianLCD, afin d'éviter tout erreur dans l'association de données. Le kidnapping peut être simplement détecté dans MonoSLAM en cherchant une variation rapide dans le nombre d'amers suivis correctement entre deux images consécutives : si le nombre d'amers suivis correctement décroît de manière conséquente d'une image à l'autre, c'est probablement parce que la caméra est obstruée ou bien qu'elle a été kidnappée (i.e., nous faisons ici l'hypothèse que lors du kidnapping, la caméra est obstruée pour masquer le trajet effectué, ou alors que le kidnapping est instantané et que le robot a pu être déplacé pendant le temps séparant deux images).

Ainsi, on peut détecter le kidnapping dans MonoSLAM en comparant la valeur actuelle du *ratio de mesures correctes*  $r_{mc}(t)$ , avec sa valeur au pas de temps précédent :

$$r_{mc}(t) = \frac{n_{\text{appariements}}(t)}{n_{\text{visibles}}(t)} \quad (11.1)$$

où  $n_{\text{appariements}}(t)$  est le nombre d'appariements entre amers projetés et primitives extraites dans l'image

à l'instant  $t$ , alors que  $n_{\text{visibles}}(t)$  est le nombre d'amers de la carte qui sont visibles depuis la pose estimée pour la caméra à l'instant  $t$ . Si ce ratio décroît fortement entre les instants  $t - 1$  et  $t$  (par exemple, si  $\frac{r_{\text{mc}}(t)}{r_{\text{mc}}(t-1)} < 0.3$ ), alors le nombre d'amers de la carte qui étaient visibles et qui ont été appariés correctement avec les primitives de l'image a diminué de manière significative entre les instants  $t - 1$  et  $t$  : un kidnapping a probablement eu lieu.

Il est important de noter ici que la méthode de détection du kidnapping est une heuristique développée spécialement dans le cadre de MonoSLAM. Toutefois, il est possible d'adapter cette heuristique à d'autres cadres de filtrage, étant donné qu'il est aisé de maintenir les quantités  $n_{\text{appariements}}(t)$  et  $n_{\text{visibles}}(t)$  au cours du temps.

Afin de simuler un kidnapping dans l'expérience, on retire simplement les images de la séquence vidéo qui correspondent à environ 30% du trajet de la caméra, alors que celle-ci est en train de retourner vers le point de départ de sa trajectoire (cf. figure 11.4). Suite au kidnapping, étant donné que le ratio d'appariements corrects a chuté, le processus d'estimation est mis en pause : les amers de la carte qui correspondent à la dernière pose estimée pour la caméra sont projetées dans l'image courante mais l'association de données n'est pas réalisée, laissant les quantités estimées à leur valeur actuelle.

Rapidement, une fermeture de boucle est détectée par l'algorithme BayesianLCD, offrant la possibilité de calculer une nouvelle pose pour la caméra, comme expliqué dans le chapitre 10 de cette partie. Cela permet alors de reprendre la procédure d'association de données, ainsi que le processus d'estimation général de MonoSLAM. Le recalage de la pose de la caméra après la détection de fermeture de boucle est illustré dans la figure 11.5. La procédure d'inflation de la covariance qui est responsable de l'association correcte de données une fois qu'une fermeture de boucle a été détectée est quant à elle illustrée dans la figure 11.6. L'incertitude associée à la pose de la caméra calculée grâce à l'algorithme de géométrie multi-vues est augmentée de manière incrémentielle, résultant en une incertitude grandissante pour les amers projetés dans l'image courante (cela est notable par la taille des ellipses augmentant avec le temps). Après chaque incrément, un ensemble conjointement compatible au sens du JCT d'appariements est recherché. Lorsqu'un tel ensemble est trouvé, l'observation des amers de la carte reconnus et appariés avec les primitives de l'image courante sont utilisés pour effectuer la mise à jour du processus d'estimation du SLAM. Par conséquent, suite à cette observation, la pose estimée pour la caméra converge vers une valeur plus cohérente et plus précise. Cela est remarquable dans la figure 11.5, étant donné que la pose de la caméra dans l'image de la rangée du bas est mise à jour et les amers projetés sont correctement associés avec des primitives de l'image courante.

Enfin, la figure 11.7 donne un aperçu de la trajectoire de la caméra à la fin de l'expérience. On peut notamment constater que lorsque la caméra est retournée dans des zones déjà explorées de l'environnement suite au kidnapping, celles-ci ont été correctement reconnues.

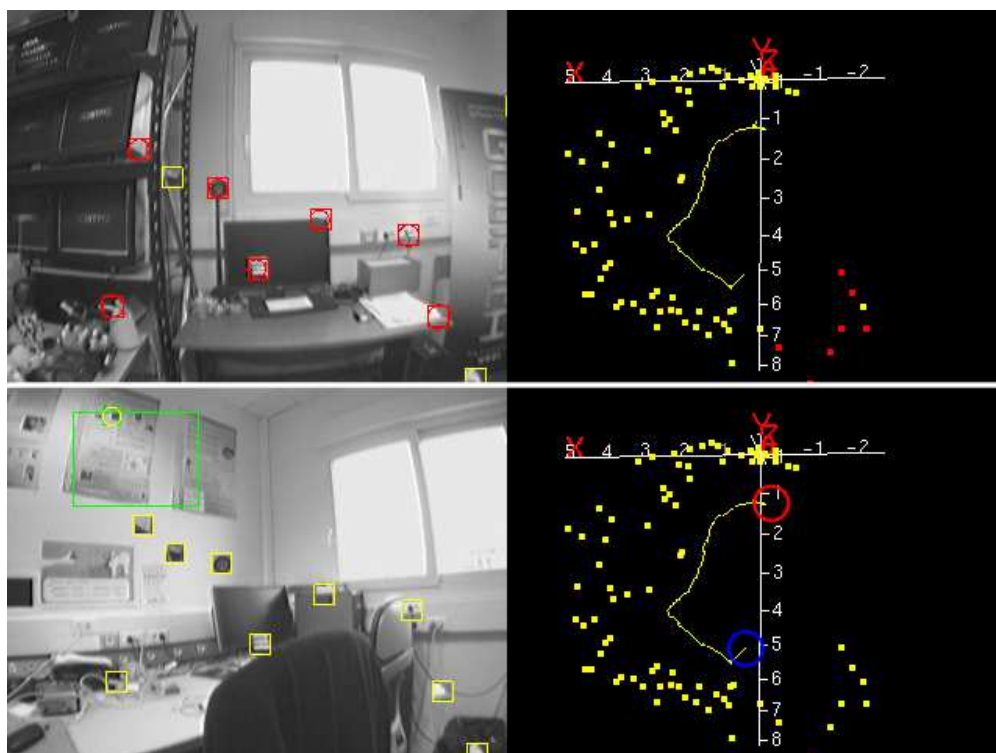


FIG. 11.4: Kidnapping : nous avons retiré les images 1500 à 1999 dans la séquence vidéo. Après ce retrait, la 1500<sup>ème</sup> de la séquence correspond en réalité à l'image numéro 2000 dans la séquence originelle : cela permet de simuler un kidnapping allant de la vue de la 1499<sup>ème</sup> image à la vue de la 2000<sup>ème</sup> image. Dans cette figure sont données la pose et la carte estimées à l'image 1499 (haut) et à l'image 1500 (bas). Les carré jaunes correspondent aux amers de la carte (lorsque dessinés dans l'image, ils correspondent aux amers visibles depuis la pose estimée). En cas d'appariement correct avec une primitive de l'image, le carré devient rouge. Si l'appariement n'a pas fonctionné, le carré est coloré en bleu. Pour tous les détails sur les conventions de couleur, le lecteur intéressé peut se référer à [Davison, 2003] et [Davison et al., 2004]. Dans la carte de l'image du bas, on donne aussi la dernière pose estimée pour la caméra (i.e., cercle bleu, avant le kidnapping), et la localisation approximative du point de vue de l'image actuellement perçue (cercle rouge, après le kidnapping).

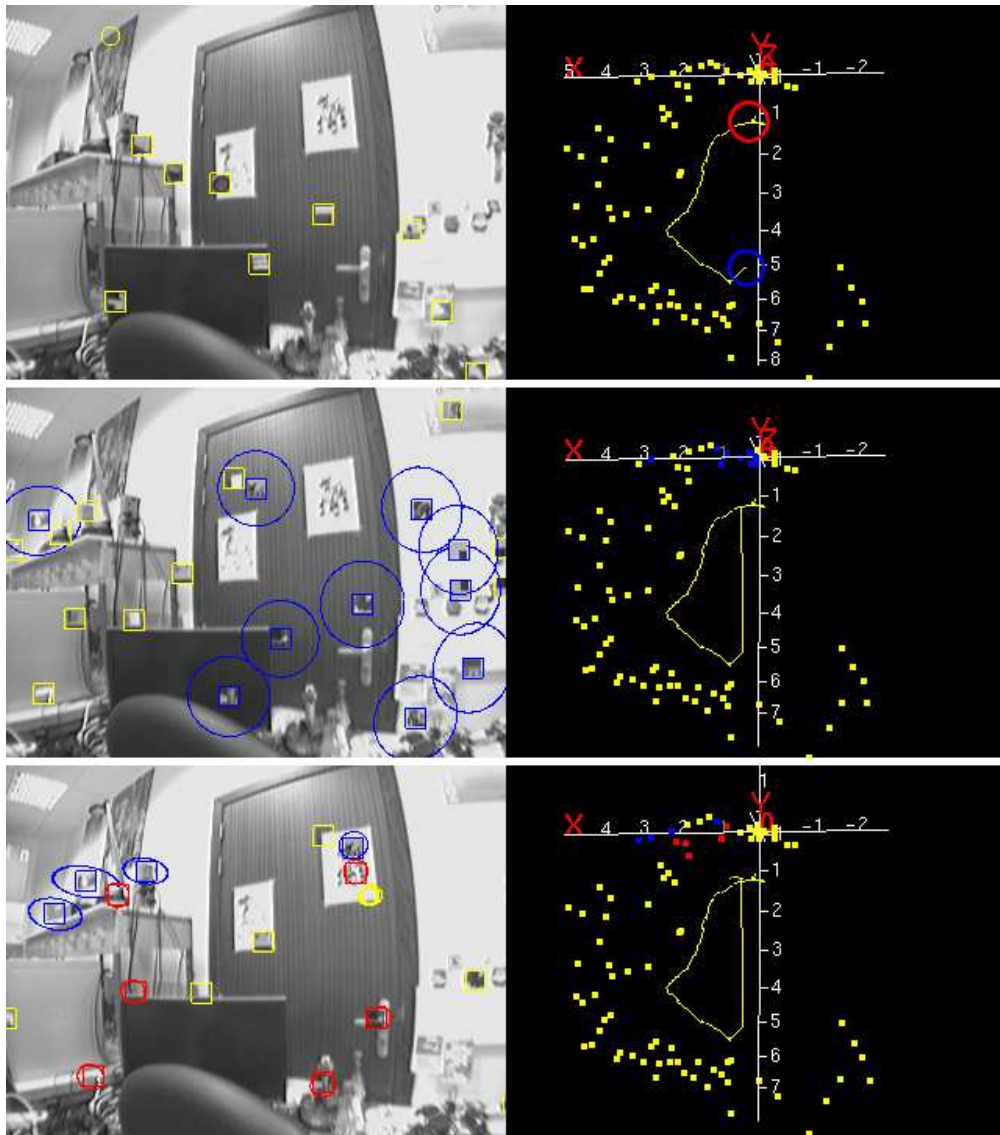


FIG. 11.5: Détection de fermeture de boucle et recalage de la pose de la caméra : lorsque la fermeture de boucle est détectée par l'algorithme BayesianLCD, une pose valide pour la caméra peut être calculée et la procédure d'association de données peut reprendre. La rangée du haut donne la pose de la caméra et la carte avant la détection de fermeture de boucle (on peut noter l'incohérente entre les amers projeté et l'image perçue). Dans la carte de cette rangée du haut sont aussi données la dernière pose estimée pour la caméra (i.e., cercle bleu, avant le kidnapping), ainsi que la localisation approximative du point de vue de l'image perçue actuellement (cercle rouge, après le kidnapping). La rangée du milieu donne la pose de la caméra corrigée grâce à la reconstruction proposée par l'algorithme de géométrie multi-vues, avec les anciens amers de la carte projetés dans l'image courante. Étant donné que ces amers ne peuvent être appariés correctement avec les primitives de l'image courante, ils sont dessinés en bleu (on remarque notamment que l'incertitude est importante, ainsi qu'indiqué par les larges ellipses). La rangée du bas présente la même situation une fois que la procédure d'inflation de la covariance a opéré (voir figure 11.6), rendant l'association de données possible : les amers de la carte appariés correctement sont maintenant colorés en rouge, et l'incertitude a chuté de manière significative (les ellipses font quasiment la même taille que les carrés représentant les primitives).



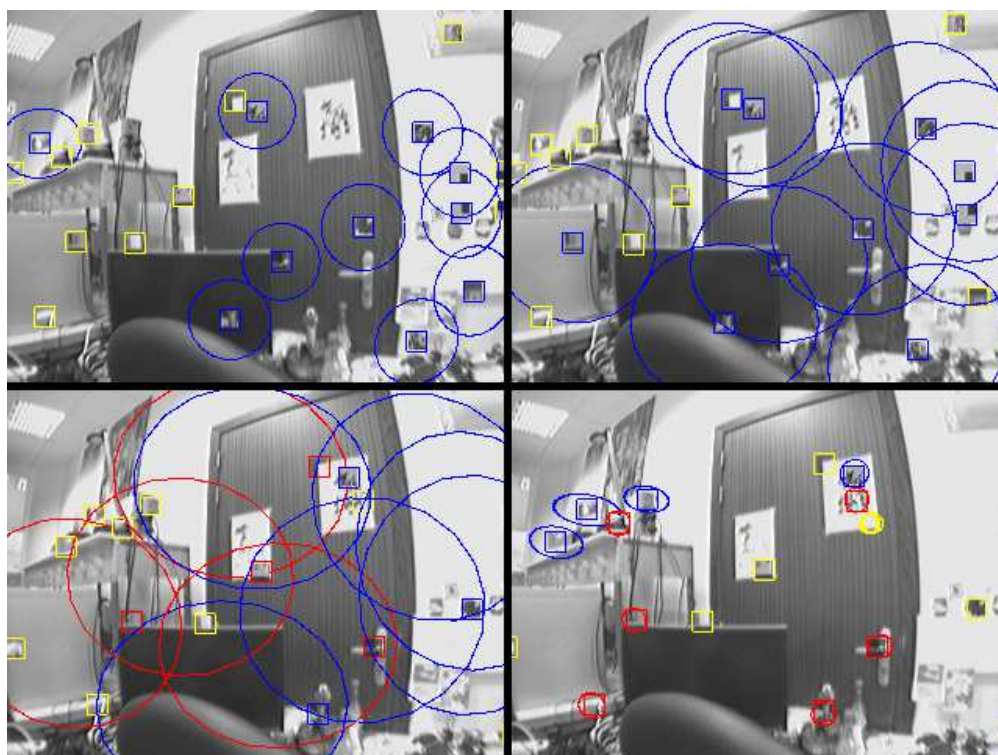


FIG. 11.6: Inflation de la covariance pour l'association de données en utilisant le JCT : du haut vers le bas, de gauche à droite, on peut voir les effets de la procédure d'inflation de la covariance, avec les ellipses d'incertitude qui grandissent jusqu'à ce qu'un ensemble conjointement compatible d'appariements soit trouvé. A ce moment, l'étape de mise à jour du FKE dans la procédure d'inférence de MonoSLAM est réalisée, résultant en une incertitude significativement plus faible (i.e., cf. l'image en bas à droite dans la figure). Ici, 4 amers sur les 10 sélectionnés ont été appariés avec des primitives de l'image (i.e., pour une mise à jour dans MonoSLAM, au plus 10 amers visibles sont sélectionnés pour l'appariement).

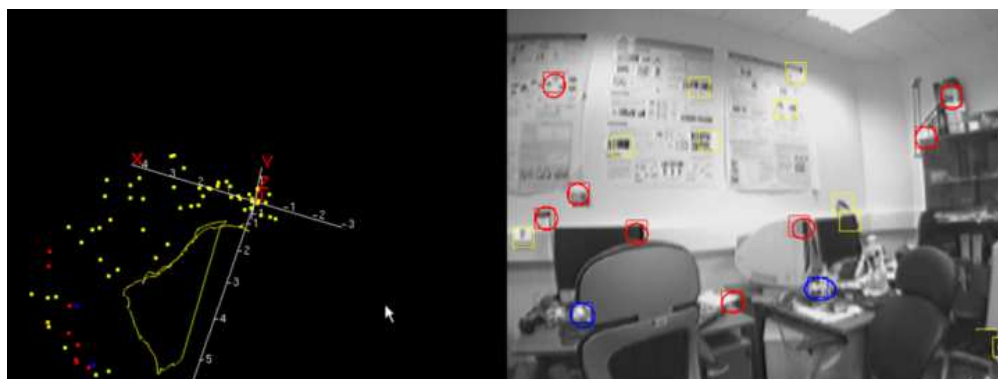


FIG. 11.7: Aperçu de la trajectoire de la caméra à la fin de l'expérience. Le cercle indique la pose finale de la caméra. On remarque que suite au kidnapping, une partie de la trajectoire parcourue à nouveau a été correctement reconnue.



## 11.3 Performances

Dans sa version originelle ([Davison, 2003], [Davison et al., 2004]), MonoSLAM permet des traitements en temps réel avec une fréquence d'acquisition des images à 30Hz. Avec la procédure d'association de données reposant sur le JCT, ce genre de performance n'est plus atteint (cf. figure 11.8) : le maintien de plusieurs hypothèses d'association de données en parallèle et leurs évaluations successives requiert d'importantes ressources en temps de calcul. Cela est principalement dû à l'implémentation naïve du JCT réalisée ici, sans chercher à l'optimiser pour atteindre des performances temps réel.

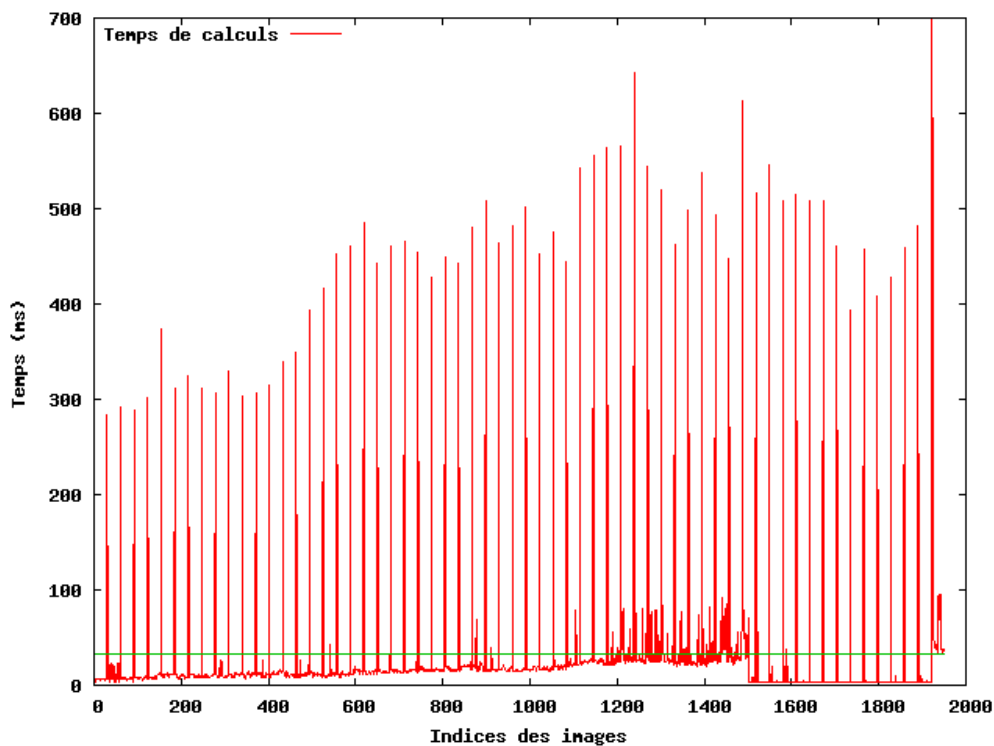


FIG. 11.8: Évolution des temps de traitements par image lorsque MonoSLAM est utilisé avec le JCT et combiné avec BayesianLCD. On aperçoit toutes les secondes un pic dû aux traitements effectués par BayesianLCD, rendant les performances temps réel impossibles à atteindre. Par ailleurs, on remarque par exemple qu'à partir de l'image 1200 environ, les temps de traitements par image dépassent les 33ms (i.e., ligne verte) requis pour atteindre des performances en temps réel (i.e., les images sont acquises à 30Hz), même lorsque BayesianLCD n'effectue aucune opération. Entre les images 1500 et 1900 environ, lors du kidnapping, le processus d'inférence de MonoSLAM est suspendu, c'est pourquoi les temps de traitements sont nuls. Le dernier pic à droite dans le graphe correspond à la fermeture de boucle, avec la procédure d'inflation de la covariance. Pour une meilleure lisibilité, l'ordonnée du graphe a été réduite, ce pic atteignant en réalité une valeur de 1.6 second environ.

Par conséquent, lorsque couplé à l'algorithme BayesianLCD pour la détection de fermeture de boucle, le système nécessite encore plus de ressources en temps de calcul, et ce particulièrement à chaque fois

qu'une image est analysée par BayesianLCD, c'est à dire toutes les secondes ici. Pourtant, nous avons limité les espaces de représentation aux primitives SIFT uniquement pour BayesianLCD, afin de restreindre les traitements, étant donné que l'environnement d'intérieur dans lequel l'expérience a été réalisée offre une structure suffisante pour que ce type de primitives soient efficaces (cf. figure 11.9 pour des exemples d'images acquises dans cet environnement d'intérieur).



FIG. 11.9: Exemples d'images composant la séquence vidéo utilisée pour l'expérience de recalage de la localisation de la caméra suite à un kidnapping.

# Chapitre 12

## Discussion

Les résultats présentés dans cette partie concernent le SLAM métrique visuel, avec l'amélioration de la procédure d'association de données de MonoSLAM grâce au Joint Compatibility Test ([Neira and Tardós, 2001]), mais également par sa combinaison à BayesianLCD pour le recalage de la localisation de la caméra suite à un kidnapping.

### 12.1 Flexibilité et adaptabilité

Nous avons cité dans l'état de l'art de cette partie (cf. page 137) plusieurs solutions ([Clemente et al., 2007], [Frintrop and Cremers, 2007], [Lemaire et al., 2007], [Se et al., 2005] et [Williams et al., 2007b]) qui sont strictement liées à un algorithme de SLAM métrique basé sur le filtre de Kalman étendu (i.e., la méthode d'inférence employée dans MonoSLAM). Nous avons déjà mentionné les problèmes que cela engendrait, et ce particulièrement au niveau du modèle de l'environnement : celui-ci n'est pas optimisé pour la tâche de détection de fermeture de boucle. Ainsi, comme nous l'avons évoqué dans le chapitre exposant notre modèle (cf. page 156), cela s'avère problématique puisque lorsque la caméra revient sur une position passée, une faible variation dans les conditions d'illumination suffit à empêcher une association de données cohérente.

Hormis son modèle de l'environnement et sa méthode d'inférence développés spécialement pour aborder la tâche de détection de fermeture de boucle, l'avantage majeur de notre solution réside dans son adaptabilité à n'importe quel algorithme de SLAM, même si celui-ci n'est pas basé sur la vision. En effet, notre système peut être perçu comme un capteur capable de fournir une mesure directe de la pose de la caméra lorsque le robot revient dans une zone qu'il a déjà cartographiée auparavant. Pour réaliser le recalage de la caméra dans ce genre de situation, nous avons mis en oeuvre un modèle d'observation spécifique, en s'attardant particulièrement sur la procédure d'association de données une fois que la pose a été grossièrement corrigée sur la base de l'information de géométrie multi-vues. Cette procédure, inspirée de [Neira and Tardós, 2001]

et couplée à une inflation incrémentielle de la covariance, offre la possibilité de prendre en compte des corrections approximatives de la pose de la caméra, même lorsque les amers de la carte sont projetés loin des primitives correspondantes dans l'image. Le JCT permet de gérer les situations ambiguës, et cela améliore grandement la précision du processus de SLAM, limitant les dérives dues aux erreurs cumulatives provenant d'imprécisions dans l'association de données.

## 12.2 Correction de la carte

Malgré la pertinence de la solution décrite dans cette partie pour le recalage de la caméra suite à un kidnapping, notre système souffre de certaines limitations qui pourraient être embarrassantes dans de simples situations de fermeture de boucle (i.e., lorsqu'un cycle est achevé dans la trajectoire du robot). En effet, dans la méthode que nous avons mis en oeuvre, la pose de la caméra est corrigée, mais la carte n'est pas modifiée : les amers qui étaient autour de la pose de la caméra avant la fermeture de boucle devraient être déplacés avec la caméra pour être disposés autour de la pose corrigée. Cela permettrait d'obtenir une carte qui soit cohérente avec la pose corrigée.

Le problème évoqué ci-dessus est dû à la perte des éléments de corrélation entre caméra et amers dans la matrice de covariance associée au vecteur d'état du filtre de Kalman étendu de MonoSLAM. Ces éléments maintiennent les relations liant l'estimation de la pose de la caméra aux estimations des poses des amers : ils sont notamment responsables de la propagation de la correction apportée à la pose de la caméra sur les positions des amers proches du point de vue correspondant. Par conséquent, lorsque ces éléments sont effacés (i.e., mis à 0), toute correction de la pose de la caméra ne pourra affecter les positions des amers de la carte.

La perte de ces éléments de corrélation dans la matrice de covariance est due à l'implémentation du modèle d'observation directe de la pose. En effet, lorsque la pose corrigée est introduite dans le FKE, l'innovation mesurée suite à l'association de données est uniquement propagée aux quantités estimées qui sont liées aux quantités observées (voir section 10.4.1). Étant donné que dans le modèle d'observation directe de la pose les quantités observées ne sont liées qu'à la pose de la caméra, et pas aux amers de la carte, la pose corrigée ne sera plus liée aux amers après la mise à jour. Par conséquent, les positions des amers ne seront pas corrigées : la caméra a été déplacée dans la carte, mais les amers restent à la même place. Cette caractéristique du modèle d'observation semble logique lorsque l'on considère l'information de mise à jour comme venant d'un capteur fournissant une observation directe de la pose : étant donné que cette information n'est pas obtenue sur la base des quantités estimées par ailleurs dans le filtre de Kalman étendu, il serait difficile de maintenir les liens existant avec ces quantités. Dans le modèle d'observation classique des amers du FKE, l'observation d'un amer lié directement à la pose de la caméra permet de corriger celle-ci, alors que les éléments de corrélation inter-amers permettent la mise à jour de la carte. Cependant, même dans le cas du modèle d'observation directe de la pose, les éléments de corrélation relatifs à la pose de la

caméra oubliés réapparaîtront dès que la caméra observera à nouveau d'anciens amers de la carte.

Les auteurs de [Williams et al., 2008] proposent une solution qui permet la correction de la carte en plus de la correction de la pose de la caméra lors d'une détection de fermeture de boucle. L'idée est de maintenir une nouvelle carte suite à la détection de fermeture de boucle, aux alentours de la pose corrigée. Cette nouvelle carte est construite indépendamment de la carte inférée jusque-là. Rapidement, une fois que cette nouvelle carte possède un nombre d'amers conséquent, elle est appariée avec la partie de l'ancienne carte correspondant à la pose de la caméra avant la détection de fermeture de boucle. Cette mise en correspondance est similaire à l'appariement de sous-cartes mis en oeuvre par les auteurs de [Clemente et al., 2007]. Ainsi, lorsque la nouvelle carte et l'ancienne peuvent être correctement alignées, elles sont fusionnées pour produire une estimation cohérente de la pose de la caméra et de la carte de l'environnement.

Il existe par ailleurs des approches alternatives au problème de la mise à jour de la carte en cas de détection de fermeture de boucle. Comme indiqué dans la section 9.4 de l'état de l'art de cette partie, il est possible de réaliser une mise à jour cohérente de l'état dans les solutions qui maintiennent une estimation de la totalité de la trajectoire, comme celles présentées dans les travaux de [Ho and Newman, 2007] et de [Newman et al., 2006]. Dans ces approches à états retardés en effet, les observations consistent en des transformations relatives entre les positions de la trajectoire. L'information correspondante provient des détections de fermeture de boucle dans l'algorithme. Ainsi, cette information peut être simplement intégrée au processus d'estimation du SLAM. De plus, dans les algorithmes de SLAM à états retardés, une carte absolue n'est pas maintenue, comme c'est le cas dans le cadre classique du FKE. A la place, chaque amer est associé à la position à partir de laquelle il a été perçu, et sa position est maintenue dans le référentiel local à ce point de vue. Par conséquent, lorsqu'un cycle est achevé, les positions des primitives n'ont pas besoin d'être mises à jour, étant donné que leurs points de vue le seront. Il est toutefois possible d'obtenir une carte absolue en transformant les positions locales des amers en positions absolues sur la base de la composition des points de vue de la trajectoire.

Les approches abordant la problématique du SLAM métrique sur la base du RBpf offrent également la possibilité de maintenir, de manière implicite, l'historique de la trajectoire du robot par le biais des poids des différentes particules. Cela permet notamment de sélectionner, lorsqu'une boucle est fermée, l'échantillon dont l'historique est cohérent avec la fermeture de boucle. Toutefois, comme mentionné dans l'état de l'art (voir section 9.3), le phénomène d'oubli, décrit dans [Bailey et al., 2006], et qui est responsable de la dégénérescence du processus d'estimation, est problématique lorsque des cycles sont achevés dans la trajectoire.

## 12.3 Perspectives

Nous avons présenté ici une solution permettant de récupérer suite à un kidnapping. Toutefois, dans notre système, BayesianLCD ne permet pas la détection du kidnapping, et cet événement doit être géré dans

le cadre de MonoSLAM. Afin d’obtenir une solution plus homogène, il serait préférable de pouvoir s’apercevoir qu’un kidnapping a eu lieu directement dans le cadre de BayesianLCD. Cela offrirait la possibilité d’adapter plus facilement cette méthode à tout algorithme de SLAM autre que MonoSLAM.

Dans le cadre du recalage de la pose de la caméra, il serait intéressant de pouvoir tirer profit des événements de “non fermeture de boucle” pour éviter des associations de données hasardeuses dans le processus de SLAM. En effet, certaines fois la pose estimée est erronée, conduisant à la projection d’amers de la carte dans l’image courante alors que ceux-ci ne sont pas visibles en réalité depuis le point de vue de la caméra. Cela peut engendrer des corrections de la pose vers une zone déjà cartographiée par le passé alors qu’aucune boucle n’a été fermée. Ce genre de situation peut être provoquée simplement par de faibles erreurs dans l’orientation de la caméra. En associant les amers de la carte aux lieux à partir desquels ils ont été perçus dans le modèle de l’environnement de BayesianLCD, on peut prédire leur occurrence dans l’image courante : les amers qui appartiennent à un lieu qui n’est pas le lieu courant (tel qu’estimé par l’algorithme BayesianLCD) ne devraient pas être pris en compte dans la procédure d’association de données.

## 12.4 Conclusion

Dans cette partie, nous avons montré une nouvelle application de notre solution de détection de fermeture de boucle en adaptant celle-ci au cadre du SLAM métrique basé sur la vision uniquement. Pour cela nous avons développé une version reposant sur la vision du Joint Compatibility Test [Neira and Tardós, 2001] spécialement pour MonoSLAM. Cela a permis d’une part d’améliorer les performances originelles de MonoSLAM, en lui conférant des capacités de détection de fermeture de boucle sans être combiné à BayesianLCD. D’autre part, cela a été indispensable à l’association de données suivant un kidnapping du robot pour pouvoir estimer précisément la pose de la caméra, une fois que celle-ci est retournée dans une zone déjà cartographiée. Pour gérer ce genre de situation, nous avons mis en place une heuristique simple pour détecter le kidnapping, et MonoSLAM a dû être associé à BayesianLCD pour reconnaître les lieux passés et pour obtenir une estimation de base de la pose corrigée grâce à l’algorithme de géométrie multi-vues.

# Conclusion

## Bilan de nos travaux

Dans ce mémoire, nous avons présenté une solution de détection de fermeture de boucle basée sur la vision uniquement et permettant des traitements incrémentiels et réalisables en temps réel. La méthode mise en oeuvre offre la possibilité de prendre en compte différents espaces de représentation de l'environnement, afin de mieux en extraire les caractéristiques pertinentes. Pour cela, les images sont encodées selon le paradigme des sacs de mots visuels, sur la base de primitives locales. Par conséquent, les lieux du modèle de l'environnement construit au fur et à mesure de l'acquisition des images sont définis grâce à ce même formalisme, par les mots visuels qui les composent. Dans les différentes expériences de fermeture de boucle réalisées notamment dans la partie I de ce mémoire, nous avons montré la robustesse de notre système face au problème de l'aliasing perceptuel : même dans des situations ambiguës où plusieurs lieux distincts se ressemblent fortement, aucun faux positif n'a été détecté.

En tirant profit de la nature discrète du modèle de l'environnement, nous avons montré qu'il était possible de convertir cet algorithme de détection de fermeture de boucle en un algorithme de SLAM topologique basé sur l'apparence uniquement. Pour cela, la décision d'ajouter ou de mettre à jour un noeud dans la carte dépend du résultat de la détection de fermeture de boucle. En définissant une simple relation d'adjacence temporelle entre les lieux traversés, nous avons fourni des résultats expérimentaux dans la partie II démontrant la qualité de notre approche : sur la base d'une simple caméra monoculaire, nous avons construit des cartes topologiques cohérentes d'environnements d'intérieur et mixte en temps réel.

Par ailleurs, l'implémentation indépendante de tout algorithme de SLAM de notre solution de détection de fermeture de boucle rend son application possible pour améliorer les performances d'algorithmes de SLAM métrique par exemple. Ainsi, dans la partie III nous avons montré comment l'algorithme BayesianLCD pouvait être couplé à MonoSLAM, un algorithme de SLAM métrique basé sur le filtre de Kalman étendu. Par le biais de cette combinaison, et en développant une version basée vision et adaptée à MonoSLAM du Joint Compatibility Test, nous avons réussi à récupérer d'une situation de kidnapping du robot, en retrouvant une position et une orientation valides pour la caméra.

D'une manière générale, les résultats présentés dans ce mémoire prouvent à quel point les solutions

indépendantes de détection de fermeture de boucle peuvent être utiles aux algorithmes de SLAM, qu'ils soient métriques ou topologiques. Nous avons évoqué, au cours des différents états de l'art réalisés ici, la progression constante des algorithmes de SLAM, ceux-ci étant capables de fonctionner sur des distances de plus en plus impressionnantes, dans des environnements de plus en plus complexes, en reposant de plus en plus sur la vision. Cependant, nous avons vu à chaque fois qu'il existait des limitations à tous ces progrès, nécessitant l'emploi de techniques avancées d'association de données par exemple, afin de réduire l'accumulation des erreurs au cours du temps. C'est précisément dans ce cadre que se place notre approche : en assurant une détection de fermeture de boucle efficace et sûre, il est possible d'améliorer encore les performances des algorithmes de SLAM.

## Perspectives

Malgré la qualité des résultats présentés tout au long de ce mémoire, et ce dans les différentes applications topologique et métrique, il reste encore certaines caractéristiques qui pourraient être améliorées pour augmenter encore l'efficacité de notre solution.

### Autres caractérisations d'image

Nous avons montré, au cours des différentes expériences rapportées dans ce mémoire, l'importance de pouvoir combiner plusieurs sources d'information pour la caractérisation des images. Dans cet état d'esprit, il serait intéressant d'étudier d'autres primitives, afin d'améliorer encore les performances de reconnaissance. Notamment, les descripteurs SURF ([Bay et al., 2006]), déjà mentionnés dans l'état de l'art de la partie I, semblent être une alternative intéressante à SIFT, principalement en raison des traitements rapides qu'ils permettent. D'autre part, les corrélogrammes [Huang et al., 1997] pourraient être utilisés en remplacement des histogrammes, en raison de leurs meilleures performances en termes de reconnaissance. Cependant, ils restent encore relativement lents à calculer sur toute une image.

### Perception active

Parmi les perspectives envisageables, il pourrait être intéressant de combiner ce système de détection de fermeture de boucle avec un mécanisme de vision active. Cela offrirait notamment l'avantage de pouvoir faire de la détection active de fermeture de boucle. Par exemple, dans les travaux de [Hubner and Malhot, 2007] et de [Stachniss et al., 2004], les lieux du modèle de l'environnement qui sont proches de la position estimée sont atteints grâce à un algorithme d'asservissement visuel. Cela améliore grandement la consistance des quantités estimées, étant donné qu'après une fermeture de boucle la précision de l'estimation est meilleure. Notamment, les auteurs de [Stachniss et al., 2004] proposent une stratégie de navigation astucieuse qui planifie des trajectoires en exerçant un compromis entre découverte de l'environnement et



confiance dans le processus d'estimation : le robot préférera retourner dans un lieu connu si possible, plutôt que de continuer l'exploration de zones inconnues s'il n'est pas assez confiant dans l'estimation de sa pose.

## **Performances**

Nous avons évoqué dans la discussion de la partie I de ce mémoire les avantages et inconvénients du modèle incrémentiel de construction du dictionnaire visuel pour l'encodage des images et des lieux. La principale limitation de notre solution de sacs de mots visuels émane de l'augmentation incessante du nombre de mots dans le dictionnaire. Pour palier ce problème, il serait envisageable d'essayer de maintenir une deuxième version du dictionnaire, en parallèle de celle déjà exploitée, en ne retenant que les mots discriminants (i.e., au sens du coefficient  $tf-idf$ ). Cela permettrait de retirer les mots qui n'apportent pas d'information intéressante pour la détection de fermeture de boucle. Lorsque ce vocabulaire optimisé contiendrait un nombre assez important de mots, ou alors lorsque le vocabulaire de départ deviendrait trop important, on pourrait arrêter la construction du vocabulaire initial et le remplacer par sa version optimisée. Une fois ce remplacement effectué, les temps de recherche des mots seraient plus faibles, résultant en une amélioration des temps de traitements, et par conséquent en une augmentation du nombre de lieux pour lequel l'algorithme de détection de fermeture de boucle fonctionne en temps réel. Après le remplacement, afin de garantir un fonctionnement optimal encore plus longtemps, la construction d'un nouveau vocabulaire sans les mots inintéressants pourrait reprendre.

## **Vision omnidirectionnelle et panoramique**

Nous avons vu dans certains des résultats présentés tout au long de ce mémoire que le formalisme des sacs de mots visuels employé ici pour encoder les images et les lieux permettait l'emploi de caméras grand-angle sans aucune perturbation. Dans le même état d'esprit, il serait intéressant de tester d'autres types de caméra, comme par exemple les caméras omnidirectionnelles ou panoramiques. Cela offrirait notamment une meilleure discrétisation de l'environnement, étant donné qu'avec ce genre de capteur, un même lieu est reconnaissable depuis des points de vues relativement éloignés. Par ailleurs, cela permettrait de résoudre le problème du passage ultérieur dans un endroit connu avec une orientation opposée. Il faudrait cependant certainement envisager des modifications sur les détecteurs de primitives pour tenir compte des déformations de l'environnement provoquées par ce type de caméra.

## **Localisation globale**

Dans l'introduction de ce mémoire, nous avons mentionné le problème de la localisation globale, en insistant sur le fait que celui-ci pouvait être abordé comme une tâche de détection de fermeture de boucle. Cela implique la possibilité de débiter une expérience de détection de fermeture de boucle en reposant sur

un modèle de l'environnement construit au préalable, afin de déterminer dans quel lieu découvert précédemment le robot se trouve actuellement. On pourrait donc reposer sur le même mécanisme de reconnaissance de lieu que pour la détection de fermeture de boucle, en imposant le fait que le robot se trouve dans un environnement connu : il ne faut pas ajouter de nouveaux lieux.

De plus, en associant le modèle de l'environnement de la détection de fermeture de boucle avec une carte métrique construite grâce à un algorithme de SLAM tel que MonoSLAM, il serait possible de maintenir une estimation de la position métrique de la caméra dans le cadre de la localisation globale.

### **Dictionnaire statique**

Au fil de ce mémoire, nous avons insisté sur les similarités entre la solution présentée ici et l'approche mise en oeuvre par les auteurs de [Cummins and Newman, 2007], où le modèle de l'environnement est appris au préalable lors d'une phase hors-ligne. Afin de mettre en correspondance plus précisément ces travaux avec les nôtres, il serait intéressant de réaliser des expériences reposant sur un dictionnaire statique. Pour cela, il faudrait apprendre ce dictionnaire au cours d'une phase préalable, et ensuite figer celui-ci (i.e., ne plus y ajouter de mots) lors la détection de fermeture de boucle.

Dans un état d'esprit similaire, et pour reprendre un des points de la discussion abordé dans la section 4.1 du chapitre 4 de la partie I, il serait intéressant de pouvoir figer le dictionnaire en ligne dans certaines circonstances. En effet, si le robot évoluait en permanence dans le même environnement, lorsque l'exploration serait complète, il serait inutile de continuer à mettre à jour le dictionnaire. En stoppant son évolution, on optimiserait les performances en termes de temps de calcul. De plus, il serait dès lors possible d'éliminer les mots non pertinents, afin d'alléger la structure du dictionnaire, conduisant à une amélioration supplémentaire des temps de traitement.

### **Variations dans l'environnement**

Parmi les solutions mentionnées dans les différents états de l'art dressés dans ce mémoire, certains auteurs ([Luo et al., 2007], [Pronobis and Caputo, 2007]) ont réalisé des expériences dans des environnements dynamiques ou soumis à des conditions d'éclairage différentes. Dans le premier cas, la structure de l'environnement est partiellement altérée (meuble déplacé) entre les différents passages du robot. Dans le second cas, ces passages sont réalisés à des périodes différentes, avec des conditions d'illumination dépendant de la météo (temps ensoleillé, nuageux, pluvieux) et de l'heure de la journée (matinée, après-midi, soirée). Pour permettre un fonctionnement optimal de notre solution, il faut en assurer la robustesse face à ce genre d'évènement : cela élargie la plage d'utilisation possible à des conditions couramment rencontrées dans tout type d'application de robotique.

## Expérience à très grande échelle

Même si à ce jour la majorité des applications de robotique restent confinées à des environnements de taille modeste, il existe déjà des expérimentations réalisées à très large échelle, comme dans le DARPA Grand Challenge<sup>1</sup>. Pour ce genre d'application, qui tendra progressivement à se généraliser, il faut considérer le problème du SLAM à des échelles pour lesquelles la dérive de l'estimation entraînera des erreurs de l'ordre du kilomètre. Cela rend les méthodes de fermeture de boucle indispensables au bon déroulement des opérations. Mais cela nécessite également de concevoir des solutions adaptées à ces larges échelles. Dans cet état d'esprit, les auteurs de [Milford and Wyeth, 2008a], [Milford and Wyeth, 2008b] ont réalisé avec succès des expériences de détection fermeture de boucle le long d'une trajectoire de plusieurs dizaines de kilomètres. Afin de tester la viabilité de notre approche face à ce type de problème, il serait intéressant de conduire le même genre d'expérience.

## Combinaison topologique / métrique

Au cours de ce mémoire, nous avons insisté sur la flexibilité de notre solution de détection de fermeture de boucle, en présentant des expériences dans les domaines métrique et topologique du SLAM. On entrevoit alors clairement la possibilité de réaliser une implémentation hybride de cette solution, mêlant approches topologique et métrique : on pourrait construire une carte topologique des lieux de l'environnement, avec une granularité telle qu'un même lieu couvre une surface assez importante, et associer une carte métrique précise à chacun des noeuds du modèle topologique. La détection de fermeture de boucle pourrait d'une part être appliquée au SLAM topologique, pour construire la représentation de haut niveau, comme décrit dans la partie II de ce mémoire. Celle-ci pourrait d'autre part être utilisée conjointement à un algorithme de SLAM métrique dans chacun des noeuds de la carte topologique, afin de déduire la pose de la caméra dans les lieux correspondants lorsque le robot y retourne, comme décrit dans la partie III de ce mémoire.

## Publications

Voici une liste des publications réalisées au cours de cette thèse. Les publications sont classées par thème.

### Détection de fermeture de boucle

1. Adrien Angeli, David Filliat, Stéphane Doncieux et Jean-Arcady Meyer  
Real-time visual loop-closure detection  
Paru dans les actes de "IEEE International Conference on Robotics and Automation (ICRA)"  
2008

---

<sup>1</sup><http://www.darpa.mil/GRANDCHALLENGE>

Dans cet article nous présentons notre solution pour la détection de fermeture de boucle en détaillant le modèle de probabilité utilisé. Nous introduisons le formalisme des sacs de mots visuels mis en oeuvre dans notre méthode, ainsi que le mécanisme de validation des hypothèses par la vérification de la géométrie épipolaire. Une expérience dans un environnement d'intérieur présentant un fort aliasing perceptuel sert à prouver la pertinence de notre solution, ainsi que sa robustesse. Nous insistons par ailleurs sur les aspects incrémentiel et temps réel de la mise en oeuvre des traitements comme étant des caractéristiques remarquables.

2. Adrien Angeli, David Filliat, Stéphane Doncieux et Jean-Arcady Meyer  
A Fast and Incremental Method for Loop-Closure Detection Using Bags of Visual Words  
Accepté pour publication dans la revue "IEEE Transactions On Robotics, Special Issue on Visual SLAM" 2008

Cet article est une extension de l'article précédent, reprenant la description de notre méthode de détection de fermeture de boucle mais en l'étendant à l'utilisation de plusieurs caractérisations d'images. Plusieurs séquences vidéo, provenant d'environnement d'intérieur et d'extérieur, sont utilisées pour démontrer la qualité de nos résultats, en insistant notamment sur l'intérêt de prendre en compte plusieurs espaces de représentation pour les images.

3. Adrien Angeli, David Filliat, Stéphane Doncieux et Jean-Arcady Meyer  
Incremental vision-based topological SLAM  
Paru dans les actes de "IEEE International Conference on Intelligent Robots and Systems (IROS)" 2008

Cet article présente une application au SLAM topologique de notre solution de détection de fermeture de boucle. Nous montrons comment la méthode de reconnaissance de lieux que nous avons développé peut être utilisée pour la construction incrémentielle de carte topologique sur la base de l'apparence uniquement. Nous insistons en particulier sur la robustesse de notre approche, et sur les traitements en temps réel qu'elle permet d'obtenir.

4. Adrien Angeli, David Filliat, Stéphane Doncieux et Jean-Arcady Meyer  
Incremental learning of an optimized visual dictionary for loop-closure detection  
Soumis pour participation à la conférence "IEEE International Conference on Robotics and Automation (ICRA)" 2009

Dans cet article, nous proposons une solution incrémentale pour limiter l'augmentation de la taille du modèle de l'environnement, et ainsi permettre un fonctionnement dans des conditions temps réel pendant une période de temps plus longue. Nous apprenons pour cela un modèle optimisé qui ne contient que les caractéristiques pertinentes et pérennes de l'environnement. Grâce à ce modèle, les temps de calcul sont grandement améliorés, alors que le taux de reconnaissance ne subit qu'une légère

dégradation.

5. Adrien Angeli, David Filliat, Stéphane Doncieux et Jean-Arcady Meyer

Visual topological SLAM and global localization

Soumis pour participation à la conférence “IEEE International Conference on Robotics and Automation (ICRA)”

2009

Cet article présente de nouvelles contributions étendant nos précédents travaux dans le cadre du SLAM topologique. Ainsi, en montant la caméra sur un robot mobile, nous tirons profit de l’information d’odométrie renseignant sur les déplacements relatifs séparant les images, afin d’estimer une position 2D absolue pour la position des noeuds : cela permet d’améliorer la cohérence de la carte construite. Nous proposons également une adaptation de notre solution de détection de fermeture de boucle au cadre plus restreint de la localisation globale, en simplifiant le modèle de probabilité sous-jacent. Ces deux améliorations ont fait l’objet d’expériences réalisées en intérieur et en extérieur, dans des environnements présentant un aliasing perceptuel important.

## SLAM métrique

1. Adrien Angeli, David Filliat, Stéphane Doncieux et Jean-Arcady Meyer

2D Simultaneous Localization And Mapping for Micro Aerial Vehicles

Paru dans les actes de “European Micro Aerial Vehicles (EMAV)”

2006

Cet article présente un algorithme de SLAM visuel basé sur le filtre de Kalman étendu pour la localisation 2D d’un drone. Afin de combiner description efficace de l’environnement et performances acceptables pour les traitements de l’image, nous proposons d’associer le descripteur robuste SIFT à de simples coins de Harris extraits rapidement. Nous présentons des résultats expérimentaux sur des séquences vidéo obtenues à partir de différentes plateformes volantes, en intérieur et en extérieur. Nous nous concentrons notamment sur le problème de la détection de fermeture de boucle et sur l’impact sur la qualité de l’estimation lorsque le drone revient dans une partie connue de l’environnement.

## Flux optique

1. Stéphane Doncieux, Adrien Angeli

Objets volants miniatures : modélisation et commande embarquée,  
chapitre Navigation des drones par flux optique

pages 305-328

Hermes-Lavoisier

2007

Dans cet ouvrage, nous avons rédigé un chapitre dédié à la présentation de diverses applications des méthodes de flux optique pour la robotique. Notamment, nous montrons de quelle manière un comportement biomimétique d'évitement d'obstacles peut-être mis en oeuvre simplement sur un robot. Nous proposons par ailleurs une méthode d'adaptation de la vitesse d'un drone dans un environnement encombré en fonction des obstacles présents dans les alentours.

# Bibliographie

- H. Andreasson, A. Treptow, and T. Duckett. Localization for mobile robots using panoramic vision, local features and particle filter. In *IEEE International Conference on Robotics and Automation*, 2005.
- A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Real-time visual loop-closure detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1842–1847, 2008a.
- A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. A fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions On Robotics, Special Issue on Visual SLAM*, 24(5) : 1027–1037, October 2008b.
- A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Incremental vision-based topological slam. In *IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems (IROS2008)*, 2008c.
- T. Bailey and H. Durrant-Whyte. Simultaneous localisation and mapping (slam) : Part ii. *IEEE Robotics and Automation Magazine*, 13(3) :108–117, 2006.
- T. Bailey, J. Nieto, and E. Nebot. Consistency of the fastslam algorithm. In *IEEE International Conference on Robotics and Automation*, 2006.
- T.D. Barfoot. Online visual motion estimation using fastslam with sift features. In *Proceedings of Intelligent Robots and Systems (IROS)*, August 2005.
- H. Bay, T. Tuytelaars, and L.V. Gool. Surf : Speeded up robust features. In *9th European Conf on Computer Vision*, 2006.
- N. Beaufort. Prototypage d’un système de navigation visuelle pour la robotique mobile. rapport de stage de fin d’étude ESIAL, 2008.
- J.L. Blanco, J.A. Fernández-Madrigal, and J. González. Toward a unified bayesian approach to hybrid metric–topological slam. *IEEE Transactions on Robotics*, 24 :2, 2008.
- O. Booij, B. Terwijn, Z. Zivkovic, and B. Kröse. Navigation using an appearance based topological map. In *IEEE International Conference on Robotics and Automation*, 2007.

- M. Bosse, P. Newman, J. Leonard, M. Soika, and W. Feiten. An atlas framework for scalable mapping. In *International Conference on Robotics and Automation*, 2003.
- B. A. Cartwright and T. S. Collet. Landmark maps for honeybees. *Biological cybernetics*, 57 :85–93, 1987.
- J. Civera, A.J. Davison, and J.M.M. Montiel. Unified inverse depth parametrization for monocular slam. In *Robotics Science and Systems*, 2006.
- L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardòs. Mapping large loops with a single hand-held camera. In *Proceedings of Robotics : Science and Systems*, 2007.
- G. Csurka, C. Dance, L. Fan, J. Williamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV04 workshop on Statistical Learning in Computer Vision*, pages 59–74, 2004.
- M. Cummins and P. Newman. Probabilistic appearance based navigation and loop closing. In *Proc. IEEE International Conference on Robotics and Automation (ICRA'07)*, 2007.
- M. Cummins and P. Newman. Accelerated appearance-only slam. In *IEEE Conference on Robotics and Automation*, 2008a.
- M. Cummins and P. Newman. Fab-map : Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27 :647–665, 2008b.
- A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Ninth IEEE International Conference on Computer Vision (ICCV'03)*, 2003.
- A.J. Davison, Y.G. Cid, and K. Nobuyuki. Real-time 3d slam with wide-angle vision. In *IAV2004 - 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, 2004.
- A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam : Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6) :1052–1067, June 2007.
- F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999a.
- F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *IEEE International Conference on Robotics and Automation*, May 1999b.
- M. W. M. Dissanayake, P. Newman, S. Clark, H. F. Durrant-White, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions On Robotics and Automation*, 17(3) :229–241, 2001.



- A. Doucet, S. Godshill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. Technical report, Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering, 1998.
- A. Doucet, N. Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, - :176–183, 2000.
- T. Duckett, S. Marsland, and J. Shapiro. Learning globally consistent maps by relaxation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3841–3846, 2000.
- D. Dufourd. *Des cartes combinatoires pour la construction automatique de modèles d’environnement par un robot mobile*. PhD thesis, Institut National Polytechnique de Toulouse, 2005.
- H. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (slam) : Part i. *IEEE Robotics and Automation Magazine*, 13(1) :99–110, 2006.
- Ethan Eade and Tom Drummond. Scalable monocular slam. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.
- A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, 3(3) : 249–265, 1987.
- P. Elinas, R. Sim, and J.J. Little. Stereo vision slam using the rao-blackwellised particle filter and a novel mixture proposal distribution. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.
- C. Estrada, J. Neira, and J. D. Tardòs. Hierarchical slam : real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4) :588–596, August 2005.
- R. Eustice, O. Pizarro, and H. Singh. Visually augmented navigation in an unstructured environment using a delayed state history. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, volume 1, pages 25–32, New Orleans, USA, April 2004.
- Ryan Eustice, Hanumant Singh, John Leonard, Matthew Walter, and Robert Ballard. Visually navigating the rms titanic with slam information filters. In *Proceedings of Robotics : Science and Systems*, Cambridge, USA, June 2005.
- D. Filliat. Robotique mobile, cours c10 - 2, 2005.
- D. Filliat. A visual bag of words method for interactive qualitative localization and mapping. In *IEEE International Conference on Robotics and Automation*, 2007.

- D. Filliat. Interactive learning of visual topological navigation. In *To appear in the proceedings of the 2008 IEEE International Conference on Intelligent Robots and Systems (IROS 2008)*, 2008.
- D. Filliat. *Cartographie et estimation globale de la position pour un robot mobile autonome (in french)*. PhD thesis, LIP6/AnimatLab, Université Pierre et Marie Curie, Paris, France, 2001. Spécialité Informatique.
- D. Filliat and J.-A. Meyer. Map-based navigation in mobile robots - I. a review of localisation strategies. *Journal of Cognitive Systems Research*, 4(4) :243–282, 2003.
- M. A. Fischler and R. C. Bolles. Random sample consensus : A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6) :381—395, 1981.
- D. Fox. *Markov Localization : A Probabilistic Framework for Mobile Robot Localization and Navigation*. PhD thesis, Institute of Computer Science III, University of Bonn, 1998.
- D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization : Efficient position estimation for mobile robots. In *Sixteenth National Conference on Artificial Intelligence (AAAI'99)*, July 1999.
- F. Fraundorfer, H. Bischof, and S. Ober. Natural, salient image patches for robot localization. *Proc. 17th International Conference on Pattern Recognition*, IV :881–884, 2004.
- F. Fraundorfer, C. Engels, and D. Nistér. Topological mapping, localization and navigation using image collections. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- S. Frintrop and A.B. Cremers. Top-down attention supports visual loop closing. In *European Conference on Mobile Robots*, 2007.
- J. Gaspar, N. Winters, and J. Santos-Victor. Vision-based navigation and environmental representations with an omni-directional camera. *IEEE Transactions on Robotics and Automation*, 16(6) :890–898, 2000.
- C. Giovannangeli, P. Gaussier, and J.P. Banquet. Robustness of visual place cells in dynamic indoor and outdoor environment. *International Journal of Advanced Robotic Systems*, 3(2) :115–124, 2006.
- T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool. Omnidirectional vision based topological navigation. *International Journal of Computer Vision*, 74(3) :219—236, 2007.
- S. Gourichon, J.-A. Meyer, and P. Pirim. Using coloured snapshots for short-range guidance in mobile robots. *International Journal of Robotics and Automation*, 17(4) :154–162, 2002.
- C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of 4th Alvey Vision Conference*, page 147–151, 1988.

- R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN : 0521540518, second edition, 2004. URL <http://www.robots.ox.ac.uk/~vgg/hzbook/>.
- K. L. Ho and P. Newman. Loop closure detection in slam by combining visual and spatial appearance. *Robotics and Autonomous Systems*, 54 :740–749, 2006.
- Kin Leong Ho and Paul Newman. Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3) :261—286, 2007.
- B.K.P. Horn. Relative orientation. *International Journal of Computer Vision*, 4 :59–78, 1990.
- J. Huang, S.-R. Kumar, M. Mitra, W. j. Zhu, and R. Zabih. Image indexing using color correlograms. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997.
- W. Hubner and H.A. Mallot. Metric embedding of view-graphs : a vision and odometry-based approach to cognitive mapping. *Autonomous Robots*, 23 :183—196, 2007.
- P. Jensfelt and S. Kristensen. Active global localisation for a mobile robot using multiple hypothesis tracking. In *IJCAI-99 Workshop on Reasoning with Uncertainty in Robot Navigation*, 1999.
- T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1) : 7–15, 1989. ISSN 0020-0190. doi : [http://dx.doi.org/10.1016/0020-0190\(89\)90102-6](http://dx.doi.org/10.1016/0020-0190(89)90102-6).
- T. Kanade, O. Amidi, and Q. Ke. Real-time and 3d vision for autonomous small and micro air vehicles. In *43rd IEEE Conference on Decision and Control (CDC 2004)*, December 2004.
- N. Karlsson, E. Di Bernardo, J. Ostrowski, L. Goncalves, P. Pirjanian, and M.E. Munich. The vslam algorithm for robust localization and mapping. In *Proceedings of International Conference on Robotics and Automation (ICRA)*, 2005.
- J. Kim and I. S. Kweon. Robust feature matching for loop closing and localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- K. Konolige. Large-scale map-making. In *National Conference on Artificial Intelligence*, 2004.
- Kurt Konolige and Motilal Agrawal. Frame-frame matching for realtime consistent visual mapping. In *IEEE International Conference on Robotics and Automation*, April 2007.
- J. Kosecká, F. Li, and X. Yang. Global localization and relative positioning based on scale-invariant keypoints. *Robotics and Autonomous Systems*, 52 :209–228, 2005.

- B.J.A. Kröse, N. Vlassis, and R. Bunschoten. Omnidirectional vision for appearance-based robot localization. In *International Workshop on Sensor Based Intelligent Robots*, 2002.
- B. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8 :47—63, 1991.
- T. Lemaire and S. Lacroix. Long term SLAM with panoramic vision. *Journal of Field Robotics*, 24(1-2) : 91–111, Jan-Feb. 2007.
- Thomas Lemaire, Cyrille Berger, Il-Kyun Jung, and Simon Lacroix. Vision-based slam : Stereo and monocular approaches. *International Journal of Computer Vision*, 74(3) :343—364, February 2007.
- V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28 :1465—1479, 2006.
- O. Linde and T. Lindeberg. Object recognition using composed receptive field histograms of higher dimensionality. In *ICPR04*, 2004.
- H. Ling and K. Okada. Diffusion distance for histogram comparison. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 246–253, 2006.
- D.G. Lowe. Distinctive image feature from scale-invariant keypoint. *International Journal of Computer Vision*, 60(2) :91–110, 2004.
- F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4) :333—349, 1997.
- B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, 1981.
- J. Luo, A. Pronobis, B. Caputo, and P. Jensfelt. Incremental learning for place recognition in dynamic environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. Image-based monte-carlo localisation with omnidirectional images. *Robotics and Autonomous Systems*, 48(1) :17–30, 2004.
- J.-A. Meyer. The animat approach to cognitive science. In H. Roitblat and J.-A. Meyer, editors, *Comparative Approaches to Cognitive Science*, pages 27–44. The MIT Press, 1995.
- J.-A. Meyer. Artificial life and the animat approach to artificial intelligence. In M. Boden, editor, *Artificial Intelligence*, pages 325–354. Academic Press, 1996.

- J.-A. Meyer. From natural to artificial life : Biomimetic mechanisms in animat designs. *Robotics and Autonomous Systems*, 22 :3–21, 1997.
- J.-A. Meyer and D. Filliat. Map-based navigation in mobile robots - II. a review of map-learning and path-planing strategies. *Journal of Cognitive Systems Research*, 4(4) :283–317, 2003.
- K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *International Conference on Computer Vision & Pattern Recognition*, volume 2, pages 257–263, June 2003.
- M. Milford, G. Wyeth, and D. Prasser. Simultaneous localisation and mapping from natural landmarks using ratslam. In *Australasian Conference on Robotics and Automation 2004*, 2004.
- M. J. Milford and G. Wyeth. Single camera vision-only slam on a suburban road network. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2008a.
- M. J. Milford and G. Wyeth. Mapping a suburb with a single camera using a biologically inspired slam system. *accepted to IEEE Transactions on Robotics Special Issue on Visual SLAM*, - :-, 2008b.
- M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using fastslam. In *IEEE International Conference on Robotics and Automation*, 2003.
- M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam : A factored solution to the simultaneous localization and mapping problem. In *AAAI National Conference on Artificial Intelligence*, 2002.
- M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM 2.0 : An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003. IJCAI.
- H. P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2) :61—74, 1988.
- P. Moutarlier and R. Chatila. Stochastic multisensory data fusion for mobile robot location and environment modeling. In *5th International Symposium on Robotics Research*, 1989.
- J. Neira and J.D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6) :890–897, 2001.
- J. Neira, J.D. Tardós, and J.A. Castellanos. Linear time vehicle relocation in slam. In *In Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, September 2003.
- P. Newman, D. Cole, and K. Ho. Outdoor slam using visual appearance and laser ranging. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

- M. E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *IEEE conference on Computer Vision and Pattern Recognition*, 2006.
- D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6) :756–777, 2004.
- D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Accepted for oral presentation at CVPR 2006*, 2006.
- D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, June 2004.
- D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1) :–, 2006.
- A. Pronobis and B. Caputo. Confidence-based cue integration for visual place recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- A. Pronobis, B. Caputo, P. Jensfelt, and H.I. Christensen. A discriminative approach to robust visual place recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.
- M. Pupilli and A. Calway. Real-time visual slam with resilience to erratic motion. In *IEEE Computer Vision and Pattern Recognition*, 2006.
- A. Ramisa, A. Tapus, R. Lopez de Mantaras, and R. Toledo. Mobile robot localization using panoramic vision and combinations of feature region detectors. In *IEEE International Conference on Robotics and Automation*, 2008.
- A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*, 22(1) :92–107, 2006.
- A. Remazeilles and F. Chaumette. Image-based robot navigation from an image memory. *Robotics and Autonomous Systems*, 55(4) :345–356, 2007.
- P. Rybski, F. Zacharias, J. Lett, O. Masoud, M. Gini, and N. Papanikolopoulos. Using visual features to build topological maps of indoor environments. In *IEEE International Conference on Robotics and Automation*, 2003.
- F. Savelli and B. Kuipers. Loop-closing and planarity in topological map-building. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.
- S. Se, D. Lowe, and J. Little. Global localization using distinctive visual features. In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

- S. Se, D.G. Lowe, and J.J. Little. Vision-based global localization and mapping for mobile robots. *IEEE Transactions on Robotics*, 21(3) :364–375, 2005.
- H. Shatkay and L.P. Kaelbling. Learning topological maps with weak local odometric information. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1997.
- R. Sim and G. Dudek. Learning and evaluating visual features for pose estimation. In *Seventh IEEE International Conference on Computer Vision*, 1999.
- R. Sim and G. Dudek. Learning generative models of invariant features. In *IEEE International Conference on Intelligent*, 2004.
- R. Sim, M. Griffin, A. Shyr, and J.J. Little. Scalable real-time vision-based slam for planetary rovers. In *Proceedings of the IEEE IROS Workshop on Robot Vision for Space Applications*, 2005.
- R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *International Joint Conference on Artificial Intelligence*, 1995.
- J. Sivic and A. Zisserman. Video google : A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision (ICCV)*, 2003.
- P. Smith, I. Reid, and A.-J. Davison. Real-time monocular slam with straight lines. In *British Machine Vision Conference (BMVC)*, 2006.
- R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *Workshop on Spatial Reasoning and Multisensor Fusion*, 1987.
- M.V. Srinivasan, J.S. Chahl, K. Weber, S. Venkatesh, M.G. Nagle, and S.W. Zhang. Robot navigation inspired by principles of insect vision. *Robotics and Autonomous Systems*, 26 :203–216, 1999.
- Cyrill Stachniss, Dirk Hähnel, and Wolfram Burgard. Exploration with active loop-closing for fastslam. In *Proceedings. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, September–October 2004.
- B. Steder, G. Grisetti, S. Grzonka, C. Stachniss, A. Rottmann, and W. Burgard. Learning maps in 3d using attitude and noisy vision sensors. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1) :11–32., 1991.
- H. Tamimi and A. Zell. Using scale space image histograms for global localization of mobile robots. In *36th International Symposium on Robotics (ISR)*, 2005.

- A. Tapus and R. Siegwart. Incremental robot mapping with fingerprints of places. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.
- S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21(4) :93–109, 2000.
- S. Thrun. Robotic mapping : A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*, pages 1–35. Morgan Kauffman, February 2002.
- S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.
- Sebastian. Thrun and Michael. Montemerlo. The graphslam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5–6) :403–429, May–June 2006.
- N. Tomatis, I. Nourbakhsh, and R. Siegwart. Hybrid simultaneous localization and map building : a natural integration of topological and metric. *Robotics and Autonomous Systems*, 44 :3–14, 2003.
- O. Trullier and J.-A. Meyer. Biomimetic navigation models and strategies in animats. In *AI Communications*, number 10, pages 79–92, 1997.
- O. Trullier, S. Wiener, A. Berthoz, and J.-A. Meyer. Biologically-based artificial navigation systems : Review and prospects. *Progress in Neurobiology*, 51 :483–544, 1997.
- S. Tully, H. Moon, D. Morales, G. Kantor, and H. Choset. Hybrid localization using the hierarchical atlas. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *IEEE International Conference on Robotics and Automation*, 2000.
- C. Valgren, T. Duckett, and A. Lilienthal. Incremental spectral clustering and its application to topological mapping. In *IEEE International Conference on Robotics and Automation*, 2007.
- J. Wang, H. Zha, and R. Cipolla. Coarse-to-fine vision-based localization by indexing scale-invariant features. *IEEE Transactions on Systems, Man, and Cybernetics*, 36(2) :413–422, April 2006.
- C. Weiss, H. Tamimi, A. Masselli, and A. Zell. A hybrid approach for vision-based outdoor robot localization using global and local image features. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.
- B. Williams, G. Klein, and Ian Reid. Real-time slam relocalisation. In *International Conference on Computer Vision*, 2007a.



- B. Williams, P. Smith, and I. Reid. Automatic relocalisation for a single-camera simultaneous localisation and mapping system. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2007b.
- B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardòs. An image-to-map loop closing method for monocular slam. In *IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems (IROS2008)*, 2008.
- J. Wolf, W. Burgard, and H. Burkhardt. Robust vision-based localization for mobile robots using an image retrieval system based on invariant features. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2002.
- J. Wolf, W. Burgard, and H. Burkhardt. Robust vision-based localization by combining an image retrieval system with monte carlo localization. *IEEE Transactions on Robotics*, 21(2) :208–216, 2005.
- Z. Zivkovic, B. Bakker, and B. Kröse. Hierarchical map building using visual landmarks and geometric constraints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.