Fusions multimodales pour la recherche d'humains par un robot mobile

Quentin Labourey

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTE UNIVERSITE GRENOBLE ALPES

Spécialité : **Mathématiques et Informatique**

Arrêté ministériel : 25 mai 2016

Présentée par

Quentin LABOUREY

Thèse dirigée par **Olivier AYCARD, Maître de conférences, UGA**
et
codirigée par **Denis PELLERIN, Professeur, UGA**

préparée au sein des laboratoires **Grenoble Images Parole Signal Automatique (GIPSA-lab)** et **Laboratoire d'Informatique de Grenoble (LIG)**
dans **l'École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique (MSTII)**

Fusions multimodales pour la recherche d'humains par un robot mobile

Thèse soutenue publiquement le **19 mai 2017**,
devant le jury composé de :

Laurent TRASSOUDAINE
Professeur, Université Clermont Auvergne, Rapporteur
Véronique CHERFAOUI
Maître de conférences, Université de Compiègne, Rapporteur
Didier COQUIN
Professeur, Université Savoie Mont Blanc, Président
Simon LACROIX
Directeur de recherche, LAAS, Toulouse, Examinateur
Olivier AYCARD
Maître de conférences, UGA, Directeur de thèse
Denis PELLERIN
Professeur, UGA, Co-directeur de thèse

Table des matières

1 Introduction **3**

1.1 Contexte 3

1.2 Objectifs de la thèse 9

1.3 Contributions 10

1.4 Plan de la thèse 11

2 Classification d'évènements sonores en environnement intérieur **13**

2.1 Introduction 13

2.2 État de l'art sur la classification d'événements sonores 15

2.3 Classification d'événements sonores en environnement intérieur à l'aide de fonctions de croyance 20

2.4 Résultats 34

2.5 Conclusion 38

3 Fusion audiovisuelle pour la détection de locuteurs successifs dans une conversation **41**

3.1 Introduction 41

3.2 État de l'art sur la fusion multimodale 43

3.3 Détection de locuteurs successifs par fusion audiovisuelle probabiliste 51

3.4 Résultats 69

3.5 Conclusion 76

4 Navigation dédiée à la détection d'humains à l'aide d'une fusion multimodale **77**

4.1 Introduction 77

4.2 État de l'art sur la navigation en robotique 78

4.3 Une recherche d'humains reposant sur une fusion multimodale 91
4.4 Résultats 106
4.5 Conclusion 124

5 Modélisation crédibiliste de l'environnement pour la navigation 125

5.1 Introduction 125
5.2 État de l'art sur le SLAM évidentiel 126
5.3 Cartographie évidentielle incluant l'être humain 134
5.4 Résultats 149
5.5 Conclusion 169

6 Conclusion générale et perspectives 171

Liste des publications 175

A Arbre cinématique complet du robot 177

B Architecture logicielle intégrale 179

Bibliographie 189

Remerciements

Il est temps de rendre à César ce qui est à César. Il est paradoxal que l'écriture d'une thèse se termine par les remerciements dûs à ceux sans qui elle n'aurait pas pu exister.

Ainsi j'aimerais tout d'abord remercier les membres de mon jury, qui ont bien voulu rapporter, examiner et évaluer mon travail : Véronique Cherfaoui, Laurent Trassoudaine, Simon Lacroix, et Didier Coquin. Merci pour ces questions et discussions enrichissantes à l'issue de la présentation.

Je dois remercier mes deux directeurs de thèse, Olivier et Denis, pour m'avoir permis de découvrir que ma passion résidait dans la robotique. Je n'oublie pas la partie "cachée" de mon encadrement : Michèle, qui m'a permis la découverte du monde dubitatif des fonctions de croyance et Catherine, que je remercie pour ses conseils lumineux lors de la rédaction (pas toujours aisée) de papiers et autres compte-rendus.

Je remercie évidemment ma famille, que j'ai pu un peu inquiéter sur la fin de ce long parcours : maman, papa, tout va bien, je mange mes légumes, et je dors la nuit. Marion, Jonathan, merci d'avoir su ce que c'est. Vous êtes les prochains et je sais que tout se passera bien pour vous !

Il est maintenant impératif de remercier ceux qui ont fait de la vie à Grenoble un plaisir quotidien, que dis-je, un paisible bonheur. Ainsi, dans un ordre d'ancienneté décroissant :

— Le nerf de la guerre : merci Lucia, sécrétaire du département DIS (même si plus vraiment maintenant) d'avoir protégé les doctorants comme tes petits poussins, et pour ces longues discussions dans ton bureau !
— Les vénérables anciens : merci à Aude et Robin (et Clothilde, parce que je veux être le premier à la citer dans des remerciements quels qu'ils soient), Raluca, Romain, Edouard, Dadouchi (l'homme imprévisible), Manu et les premiers co-bureaux : Naxxo, Fatima, Wei, Chengfang
— Les un peu moins vieux mais pas trop quand même : merci à Cindy (la dame de haute savoie), Benoit O., Guillaume, Arnaud (et Nestor, parce que je veux être le premier à le citer dans des remerciements quels qu'ils soient !), Pascal (les soirées à jouer jusqu'à pas d'heure), Céline, Tim G (pour ces discussions enrichissantes et pleines de sens)
— Les pairs qui se serrent les coudes : merci à Lucas (il n'y a qu'ici que je m'excuserai pour ces 45 minutes de violence fortuite avec un coussin), Tim 2G (dit "le magnifique", mais dans "victime", il y a "Tim"), Taia l'embrouille, Emmanuelle (la même crise de nerfs que moi), Mael et Maelle, Nelson (le colloc fantôme), Renaud (sur qui je peux compter à n'importe quelle heure du jour mais surtout de la nuit), Marine. Enfin un merci spécial à Benoit A. (et à Sylvia qui a fait de toi un meilleur homme) : après la nuit, avant le jour, je t'offrirai les hautes lumières. Bonsoir.
— La relève : merci à Marc, Pierre (merci d'avoir souffert pour apparaître dans ma soutenance), Antony et Alexandre, les 4 mousquetaires du GIPSA, à Miguel (ne deviens

jamais PDG tu fais trop peur), Florent. Merci du fond du coeur à Raphaël, je n'aurais pas survécu dans ces mois difficiles sans ta gentillesse, ta patience et ton amitié, tu as été au top.

— Les p'tits nouveaux : merci à Marion, Marielle (encore plein de musique à découvrir et de beaux endroits à visiter !), Camille, Kévin, Pedro. Merci à Katia pour cette sagesse matinale lors de la préparation de mon oral (petit escargot, porte sur son dos, sa maisonnette).
— Les arrivés trop tard : merci à Victor (et à ses Manhattan de renommée internationale), Mélisande (et les jolis morceaux de piano, et gnagnagba), Paolo (a la tua m... sono divantato immune).

Merci enfin à toi Céline, pour ta patience dans ces moments difficiles. Merci d'avoir partagé mon quotidien au travers de cette thèse et bien plus longtemps encore.

Chapitre 1

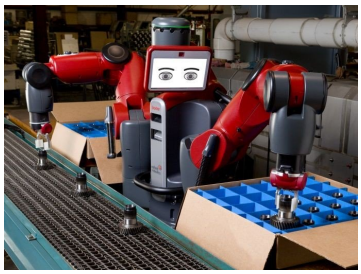
Introduction

1.1 Contexte

La robotique est une thématique largement explorée dans les livres, les films, et l'imaginaire collectif depuis des années. On y voit souvent des robots dotés de capacités quasi-similaires, voire supérieures aux êtres humains qu'ils côtoient au quotidien et avec qui ils vivent. Bien que les robots performants présentés par ces oeuvres de science-fiction n'existent pas à l'heure actuelle, la robotique est un domaine de recherches qui a beaucoup progressé au cours des dernières décennies. Les domaines d'application concernés sont variés (figure 1.1), allant de la médecine (figure 1.1a) aux véhicules autonomes (figure 1.1b), en passant par les usines de production (figure 1.1c), et les systèmes ménagers (figure 1.1d).

(a) DaVinci, robot médical

(b) La google car, voiture autonome



(c) Baxter, le robot producteur

(d) Roomba, petit aspirateur autonome

Figure 1.1 – Quelques exemples de robots

La fédération internationale de la robotique (International Federation of Robotics, IFR [1]) [Ifr] divise les robots en deux grandes catégories (figure 1.2) :

— Les robots industriels, qui ont pour but d'automatiser des tâches. Ils sont principalement utilisés dans des chaînes de production (par exemple, en tant que robots d'assemblage). Des robots industriels capables d'apprendre un geste ou une tâche en observant un humain ont commencé à apparaître. C'est le cas de Baxter, qu'on peut voir à la figure 1.1c.

— Les robots dits "de services", qui ont pour but d'effectuer des tâches utiles pour les humains (à l'exception de tâches industrielles). Les robots de services sont divisés en deux sous-catégories : les robots de services professionnels qui sont contrôlés par des utilisateurs entraînés dans le cadre de leur métier (figure 1.1a), et les robots de services personnels domestiques qui sont utilisés par des personnes non-initiées au maniement de robots, dans leur vie courante (figure 1.1d).

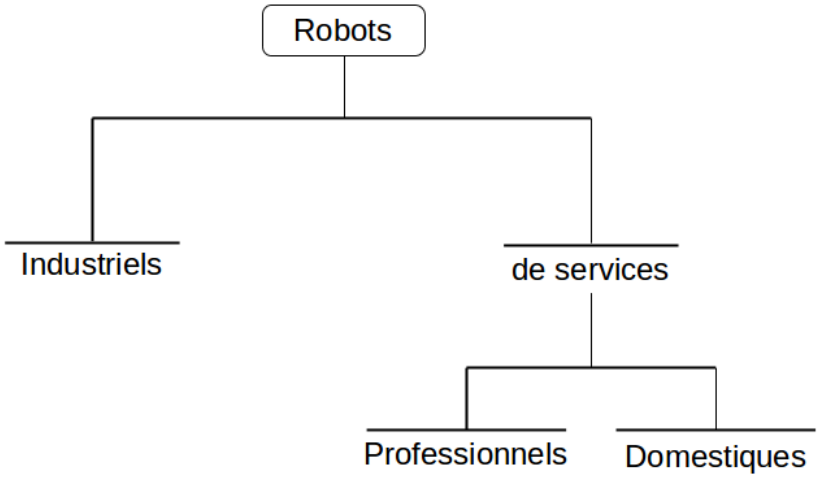


FIGURE 1.2 – Les robots peuvent se différencier en sous-catégories, d'après [Ifr]

La diminution du coût général de l'électronique (capteurs, moteurs, processeurs...), liée à la croissance de la puissance de calcul qu'il est possible d'embarquer dans un robot de taille modérée, amène naturellement à penser que les robots sont destinés à vivre avec les êtres humains. Ces facteurs conjugués ont fait croître le marché de la robotique de services personnels, provoquant l'intérêt du grand public pour les robots. Ainsi les prévisions de vente de robots domestiques pour les années 2015-2018 projetées par l'IFR montrent une augmentation sensible (figure 1.3), de 4.7 millions d'unités vendues en 2014 à une prévision de 25.2 millions d'unités vendues sur la période 2015-2018 (soit une moyenne de 8.4 millions d'unités vendues par an).

Les robots compagnons

Les robots domestiques destinés au grand public actuellement disponibles sont dans la majorité des cas des robots ménagers (aspirateurs, tondeuses à gazon, nettoyeurs de vitres, etc...) ou des jouets. Ils sont utilisés comme des outils à l'intelligence limitée, et dédiés à une tâche

1. www.ifr.org

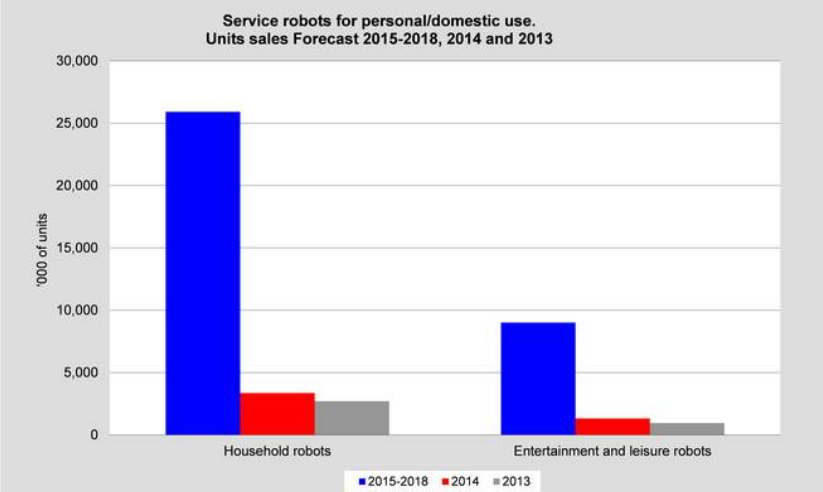
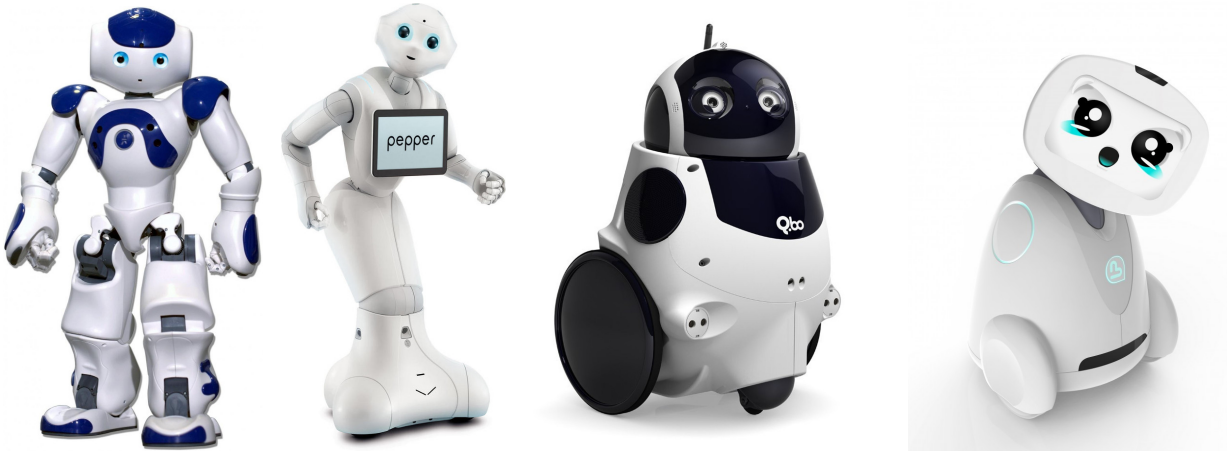


FIGURE 1.3 – Prévisions de ventes de robots de services à usage personnel projetées par la fédération internationale de la robotique, extraites de [Ifr]

précise. Cependant une nouvelle sorte de robots de services personnels commence à émerger : les robots assistants, ou robots compagnons (figure 1.4). Ces robots ont des capacités et des missions différentes des robots ménagers. Tout d'abord, ils sont destinés à l'interaction avec l'être humain. Ceci rend leur apparence importante, comme le montre [Wal+08]. Ainsi un robot compagnon aura souvent une apparence humanoïde, ou du moins une forme à laquelle un humain puisse facilement s'identifier : une tête bien visible, contenant des organes assimilables à ceux des humains (yeux, bouche), capable de représenter des émotions basiques (plissement des yeux, sourire, etc...), et un corps identifiable auquel sont rattachés des moyens de locomotion (jambes ou roues), et parfois de préhension (bras).

Les tâches attribuées aux robots compagnons ne sont pas clairement délimitées : [Dau+05] définit un robot compagnon comme un robot destiné à l'interaction sociale avec l'homme, capable d'effectuer une grande quantité de tâches telles que proposer des exercices éducatifs à destination des enfants, s'occuper de la sécurité de la maison, des tâches de la vie courante (*e.g.* promener un chien, vérifier qu'une personne est présente à un endroit, etc...), divertir les personnes présentes, délivrer des messages, etc... Cependant tous les robots compagnons ont le point commun de placer l'humain au centre de leur perception et de leurs actions. Ils doivent inclure naturellement l'humain dans leur représentation du monde pour pouvoir remplir leur mission. Qui plus est, ces robots doivent pouvoir agir de manière autonome, et naviguer de façon à ne pas gêner les humains en présence.

De tels robots, possédant l'intégralité des capacités citées ci-dessus, ne sont pas encore accessibles au grand public. L'IFR prévoit que la part de robots compagnons dans les ventes de robots domestiques sera de seulement 8100 unités vendues à travers le monde dans la période 2015-2018 (sur les 25.2 millions d'unités vendues au total, soit environ 0.32%). Des efforts



(a) Nao (b) Pepper (c) Q.bo (d) Buddy

FIGURE 1.4 – Quelques robots dits "compagnons" : (a) Nao (Aldebaran), (b) Pepper (Aldebaran), (c) Q.bo (TheCorpora), (d) Buddy (Blue Frog Robotics)

ont cependant été faits dans cette direction par les acteurs industriels de la robotique, et on voit naître des exemples d'application : l'entreprise Aldebaran, un des leaders industriels de la robotique humanoïde mondiale, se spécialise dans les robots mobiles destinés à l'interaction avec l'humain. Cette entreprise a ainsi créé Nao, petit robot humanoïde mobile (figure 1.4a), qui a été amené en maison de retraite pour proposer des animations basées sur des chorégraphies pré-programmées, ou Pepper (figure 1.4b), robot mobile qui a été utilisé par une banque japonaise pour renseigner ses clients. Q.bo (figure 1.4c), produit par la compagnie espagnole TheCorpora, est un petit robot mobile équipé d'une tête humanoïde, disponible pour tous, mais uniquement équipé nativement de fonctions très basiques lui permettant principalement de se mouvoir. Buddy (figure 1.4d), qui sera mis en vente en 2017, aura la possibilité de jouer à cache-cache avec des enfants. Ces robots sont souvent "open-source" (matériel et/ou logiciel) et disposent d'une architecture programmable adaptée à la robotique. Cette accessibilité les rend particulièrement attrayants pour la recherche : elle permet aux utilisateurs de définir eux-mêmes leur cahier des charges, avec pour seules limites celles physiques du robot.

On peut diviser la mission d'un robot compagnon en deux parties distinctes : la partie physique, qui contient toutes les stratégies de mouvement et de navigation du robot (où aller, comment patrouiller, quelle tâche réaliser, comment se positionner pour préparer une interaction, quelles données inclure dans la représentation de l'environnement, etc...), et la partie cognitive, qui relève entièrement de l'interaction avec la personne (comment savoir quand engager une interaction, puis une fois cette interaction entamée, comment la mener, observer les réactions et comprendre les intentions et les besoins des personnes, etc...).

Bien qu'un robot assistant "complet" ne soit pas disponible actuellement, les robots compagnons ont fait l'objet de plusieurs travaux de recherche dans les années récentes. Un domaine qui soulève particulièrement l'interêt de la recherche est la surveillance de personnes.

Les personnes agées, ou en situation de fragilité, pourraient être des bénéficiaires de tels robots. En effet, comme le souligne [Wal+13], la diminution du taux de natalité, conjuguée à l'augmentation de la durée de vie moyenne, provoque un phénomène appelé vieillissement démographique : en France, l'Institut National de la Statistique et des Études Économiques (INSEE [2]) prévoit qu'en 2050, un habitant sur trois sera âgé de 60 ans ou plus, contre un sur cinq en 2005 [Ins]. Cela va conduire à une augmentation du besoin en aide à la personne. Pour faire face à l'insuffisance du nombre d'aidants, une solution est l'assistance automatisée : elle est possible via des capteurs placés dans l'infrastructure des bâtiments (caméras de vidéosurveillance, microphones, etc..), mais ces capteurs sont intrusifs et cette surveillance permanente (parfois non-nécessaire) peut être mal vécue par les personnes.

En revanche un petit robot compagnon mobile bien visible patrouillant parmi les humains et s'enquérant de leurs besoins serait plus facilement accepté par les personnes, comme le montre [Wal+13]. Un scénario typique que l'on pourrait imaginer pour ce genre de robots compagnons est le suivant : un robot mobile entre dans la salle commune d'une maison de retraite où se trouvent différentes personnes. Il a pour objet de rappeler à une personne (appelons la P1) qu'elle doit prendre un médicament, s'il la croise. Il entame donc sa patrouille à la recherche de P1, en faisant attention de ne heurter ni de gêner le chemin de personne. En chemin, une personne P2 fait un signe pour attirer son attention, indiquant ainsi le besoin d'engager une interaction. Le robot décide donc d'aller s'enquérir des besoins de P2. Après avoir effectué cette tâche, le robot reprend sa patrouille de recherche de P1.

A nouveau ici, on peut différencier la partie physique (patrouille, stratégie de recherche de P1, déplacement en environnement humain, positionnement par rapport à P2), et la partie cognitive (détection d'une invitation à l'interaction, engagement et maintien d'une interaction naturelle, compréhension des besoins d'une personne). Une grande partie des travaux effectués jusqu'à présent porte sur l'aspect cognitif et l'interaction homme-robot, ainsi que le ressenti des personnes face à des robots assistants. [BHR09] propose une revue des effets de différents robots compagnons, basée sur 5 critères : amélioration de l'état de santé de la personne, de l'humeur, facilité de communication, combat de la solitude, ou autre critère. Depuis ce travail, d'autres études ont eu lieu sur les différents aspects de l'interaction Homme-Robot : [VJC16] propose par exemple une méthode de détection de signes montrant la volonté d'un humain d'engager une relation sociale avec le robot.

Jusqu'à récemment, peu de travaux étaient centrés sur la partie physique de tels robots, comme le souligne [Spa15] : bien que la navigation de robots ne soit pas un sujet nouveau [TBF05], les travaux portant sur la navigation en environnement dynamique humain, avec une représentation du monde centrée sur l'humain, commencent à peine à apparaître.

Architecture classique d'un système robotique

Dans son livre *Probabilistic Robotics* [TBF05], Sebastian Thrun définit la robotique comme la science de percevoir et agir sur le monde réel au travers d'un objet mécanique contrôlé par

2. www.insee.fr

ordinateur. Ainsi le fonctionnement d'un robot mobile autonome est classiquement caractérisé par le triptyque "Perception-Décision-Action" (figure 1.5).

Ce triptyque se décline de la manière suivante :

— Perception : traitement de flux de capteurs extéroceptifs pour extraire de l'information sur l'environnement (traitement d'image, analyse de données laser, traitement audio, fusion de données capteurs). Le type de traitement effectué sur ces flux dépend de la mission imposée au robot, et vise à extraire de l'information pertinente pour la remplir.

— Décision : à partir de l'information extraite, et en prenant en compte l'objectif du robot, ainsi que ses possibilités de déplacement, le robot doit décider d'une stratégie d'action (choix du prochain objectif, planification de trajectoire, planification de tâches).

— Action : une fois la stratégie mise en place, le robot réalise la commande nécessaire pour remplir les objectifs fixés lors de l'étape de décision, influant ainsi sur l'état de son environnement.

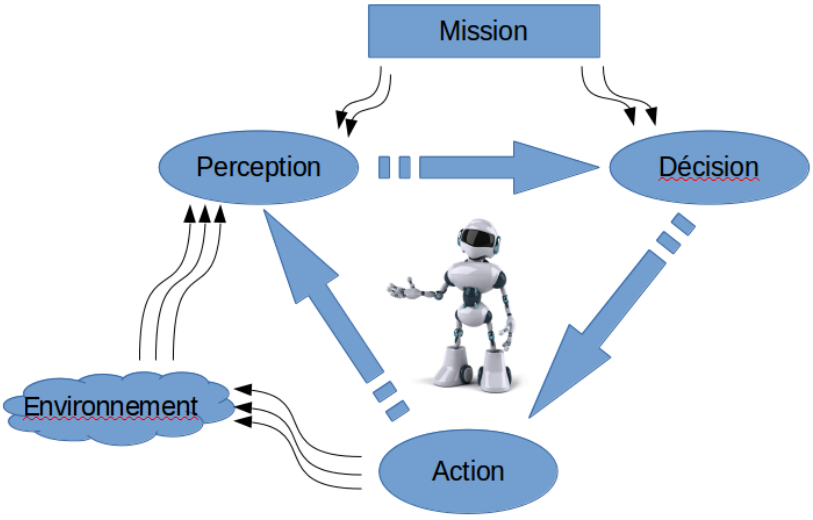


FIGURE 1.5 – Représentation du cycle classique de la robotique : perception, décision, action

Ainsi une architecture robotique devra exploiter chacun des éléments de ce triptyque. Les architectures ont donc souvent la forme générale représentée par la figure 1.6. La fusion de données extraites des capteurs prend un rôle important dans ce genre d'architecture, car elle déterminera les éléments pertinents que le robot inclura dans la représentation de son environnement.

C'est dans le contexte des robots compagnons que se place cette thèse, qui vise à proposer des méthodes de perception et de navigation, exploitant le trio "perception-décision-action" pour le déplacement d'un robot mobile dans un environnement dynamique composé d'humains. Plus précisément, nous nous concentrerons sur la surveillance d'êtres humains dans un environnement intérieur. Chacune des parties du schéma de la figure 1.6 sera étudiée et des contributions seront apportées à chacun des niveaux.

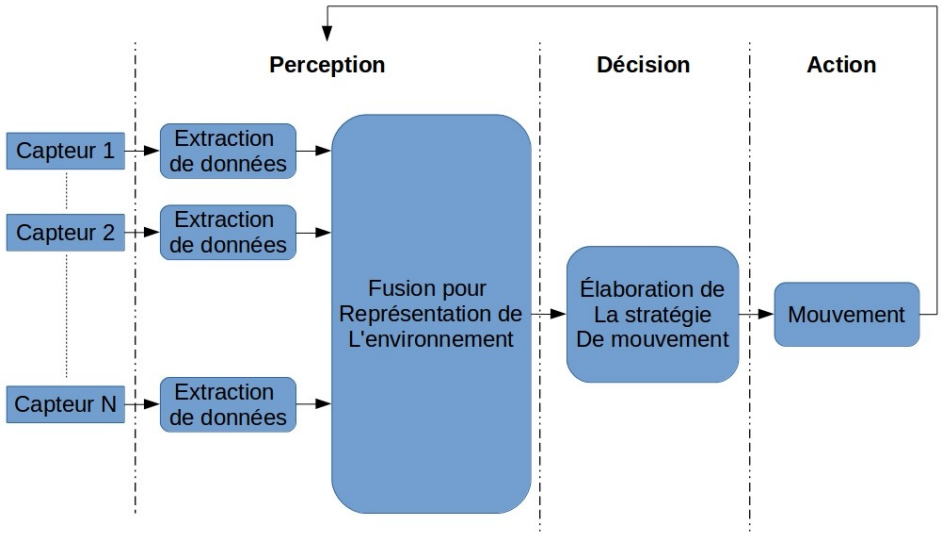


FIGURE 1.6 – Architecture logicielle d'un système robotique

1.2 Objectifs de la thèse

Nous considérons le cas d'un robot mobile d'intérieur dont l'objectif est de détecter les humains présents dans l'environnement et de se positionner physiquement par rapport à eux, dans le but ultérieur d'engager une interaction sociale. Ce robot dispose pour cela de deux types de capteurs : un capteur RGB-Depth de type kinect®, et des microphones. Il dispose aussi d'actionneurs lui permettant de se mouvoir dans son environnement. Du point de vue du robot, une telle mission implique plusieurs éléments : tout d'abord, il est nécessaire que le robot soit capable de se localiser dans son environnement. Cela commence généralement par une étape de cartographie, qui permet de représenter l'environnement statique dans lequel se déplacera le robot (meubles, murs...). Une fois cette carte établie, des méthodes de localisation peuvent être utilisées par le robot. Le robot devra par la suite détecter les êtres humains de son environnement, bien sûr, mais aussi les localiser dans l'espace qui l'entoure. Pour ce faire, ce robot utilise ses capteurs, lui permettant de percevoir son environnement, à travers des données hétérogènes et parfois incertaines. En effet, les capacités de perception du robot sont limitées tant par les contraintes purement matérielles (limites de mesures, résolution, distance maximum de perception...) que par les limites des traitements proposés (fausses détections, fausses alarmes, etc..). Le robot doit être capable d'inclure ces incertitudes dans sa représentation de son environnement. Cette représentation doit aussi être le lien avec la partie navigation : une fois les personnes détectées, le robot doit choisir l'endroit où il souhaite se positionner pour être en mesure d'établir une interaction avec la personne choisie le plus facilement possible, *i.e.* de manière à être bien visible par la personne en question.

Ainsi cette thèse comporte plusieurs objectifs :

— Étudier des algorithmes de perception permettant d'extraire l'information des flux cap-

teurs. Les modalités utilisées apportant chacunes des informations de natures différentes, cet objectif peut se décliner en deux sous-catégories : la perception unimodale, concentrée sur la proposition de méthodes permettant l'extraction d'informations pertinentes dans une modalité précise (détection visuelle, classification de sons), et la perception multimodale, basée sur la fusion de données capteurs permettant de prendre en compte l'information fournie par plusieurs types de capteurs.
— Proposer une représentation de l'environnement du robot adaptée à la mission et à la mobilité du robot, ainsi qu'à ses caractéristiques physiques. Cette représentation devra être en mesure d'inclure les informations fournies (*i.e.* les endroits où se trouvent les humains) par les modalités utilisées ainsi que l'incertitude qui leur est liée. Elle reposera sur une fusion des informations capteurs.
— Implémenter une stratégie de navigation pour permettre à un robot réel de naviguer dans un environnement dynamique composé d'humains et de se positionner par rapport aux humains présents.
— Évaluer les algorithmes proposés. Ceci pourra être divisée en deux catégories : tout d'abord la simulation, qui permettra une preuve de concept des méthodes. Une simulation étant intrinsèquement limitée par le nombre de paramètres simulés, elle permettra généralement de démontrer leur utilité et leur fonctionnement. La deuxième catégorie est l'implémentation sur un robot réel, et le test dans des environnements réalistes contenant des humains. On pourra ainsi se confronter aux limites réelles de la perception du robot et de la représentation du monde proposée.

1.3 Contributions

Cette thèse comporte des contributions de différentes natures :
— **Classification d'événements sonores en environnement intérieur :** Nous avons proposé un système de classification de sons, en utilisant un petit nombre de classes (parole, musique, impacts et sons de l'environnement). Cette classification, basée sur la théorie des fonctions de croyance, a l'avantage d'inclure une classe de doute, qui permet au robot de labelliser certains sons comme "inconnus", lui évitant de commettre des erreurs pouvant mener à la confusion. Cette méthode a été testée sur une base de sons que nous avons construite à partir de différentes bases accessibles sur internet.
— **Fusion audiovisuelle pour la détection de locuteurs successifs dans une conversation :** Ce système fusionne l'information de couleur de peau, de détection de visages, et de localisation de source sonore à l'aide d'un filtre probabiliste temporel. Ceci permet la détection en temps réel de locuteurs face au robot immobile. Cette méthode a été testée sur des vidéos acquises par le robot.
— **Navigation dédiée à la détection d'humains à l'aide d'une fusion multimodale :** A partir des informations provenant des capteurs laser, audio, vidéo et profondeur, le robot cherche des humains de manière autonome, dans un environnement connu. Ce système de perception-décision-action combine des algorithmes de perception issus de l'état de l'art. L'information obtenue des capteurs est agrégée dans des cartes de perception, qui sont fusionnées de manière simple. Le robot choisit son itiné-

raire à partir de ces cartes de perception en suivant un automate de décision. Une fois qu'un humain est visuellement détecté, le robot se place dans la mesure du possible à un mètre de lui, face à lui. La navigation se fait grâce à un diagramme de Voronoï pour permettre au robot de rester le plus loin possible des obstacles et être bien visible par les humains lorsqu'il se déplace. Ce système a été implémenté dans un simulateur dans un premier temps, puis sur un robot Q.bo. Il a été testé dans des conditions réelles, au sein d'un petit appartement.

— **Modélisation crédibiliste de l'environnement pour la navigation :** Afin d'améliorer le système précédent, nous avons proposé d'utiliser le formalisme des fonctions de croyance en ajoutant des mécanismes de fusion et de conservation de l'information dans le temps. Cela permet au robot d'avoir une représentation de son environnement comprenant l'information perçue et son incertitude, à l'instant présent, ainsi qu'aux instants précédents. Ce système a d'abord été évalué dans des simulations réalistes, puis mis en place et évalué sur le robot Q.bo.

1.4 Plan de la thèse

La manière dont les chapitres de cette thèse s'articulent autour du thème de la robotique est illustrée à la figure 1.7. Le chapitre 2 détaille le travail réalisé sur la classification d'événements sonores en environnement intérieur. Le chapitre 3 présente la fusion de données audiovisuelles pour la détection de locuteurs successifs au sein d'une conversation. Les chapitres 4 et 5 se concentrent sur la recherche d'humains par un robot mobile. Le chapitre 4 décrit un système reposant sur une fusion simple de plusieurs modalités, ainsi que son implémentation sur un robot Q.bo. Le chapitre 5 enrichit la représentation du monde proposée dans le chapitre 4 en y ajoutant le formalisme des fonctions de croyances, permettant ainsi au robot d'inclure l'incertitude des mesures capteurs dans sa représentation. Ce nouveau formalisme, plus performant, a tout d'abord été implémenté en simulation puis testé en conditions réelles sur le robot Q.bo. Finalement, le chapitre 6 récapitule les contributions de cette thèse et les perspectives ouvertes par ces travaux.

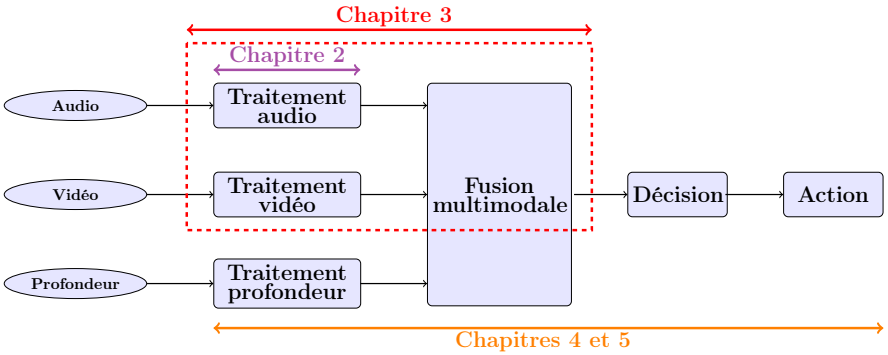


FIGURE 1.7 – Illustration du plan de thèse

Chapitre 2

Classification d'évènements sonores en environnement intérieur

Sommaire

2.1 Introduction 13
2.2 État de l'art sur la classification d'événements sonores 15
2.2.1 Définition du problème de classification 15
2.2.2 Présentation des travaux existants 17
2.2.3 Synthèse de l'état de l'art 20
2.3 Classification d'événements sonores en environnement intérieur à l'aide de fonctions de croyance 20
2.3.1 Taxonomie choisie 21
2.3.2 Méthode de classification 23
2.3.3 Caractéristiques étudiées 32
2.4 Résultats 34
2.4.1 Base de sons construite 34
2.4.2 Sélection et performance des caractéristiques 35
2.4.3 Résultats de classification 37
2.5 Conclusion 38

2.1 Introduction

Ce chapitre décrit un travail qui s'inscrit dans le cadre de la perception. L'audio est une modalité encore peu explorée dans le domaine de la robotique. Or, dans le cas d'un robot compagnon, cette modalité contient des informations pertinentes pour guider un robot, telles que des marqueurs de présence humaine. Ce sont ces marqueurs que l'on va chercher ici à classifier.

L'être humain a une capacité naturelle à différencier et reconnaître les sons ambiants de manière quasi-instantanée. Le son joue un rôle fondamental dans sa façon d'agir. Des travaux ont montré [SPG12 ; SPG13] que les êtres humains ne réagissaient pas de la même manière lorsqu'ils visionnent un film sans ou avec une bande son. Naturellement, les humains extraient

deux informations du son entendu : le contenu sémantique ("Qu'ai-je entendu ?") du son et sa localisation ("D'où ce son venait-il ?").

De manière similaire, l'audio peut receler des informations importantes pour un robot destiné à surveiller les humains. En considérant les informations de localisation et de contenu, le robot peut décider ou non d'aller explorer plus en détails certains endroits pour trouver la source du son entendu. De plus, certains sons seront prioritaires sur d'autres, e.g. il est plus intéressant d'aller enquêter sur la source d'un son caractérisé comme étant de la parole que sur un son caractérisé comme de la musique.

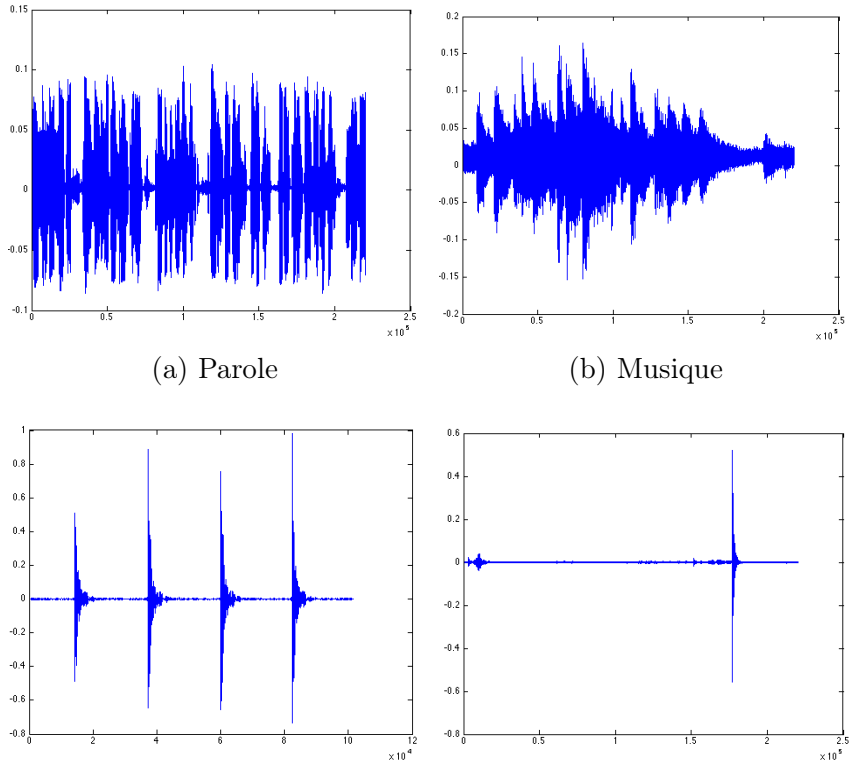
Pour illustrer ceci, reprenons l'exemple proposé dans le chapitre précédent : un robot entre dans la salle commune d'une maison de retraite où se trouvent différentes personnes. Il a pour mission d'aller voir une personne P1. Au moment où le robot pénètre dans la salle commune, aucun de ses capteurs n'est en mesure de l'informer sur la présence possible d'humains à certains endroits. Soudainement, une personne se met à parler sur la gauche du robot, en dehors du champ de ses autres capteurs. Il semble donc normal pour le robot d'aller voir si la personne qui parle est P1, en l'absence d'autres informations. A présent, imaginons qu'en chemin, un autre patient chute en se cognant violemment par terre. Le robot dont la mission globale est toujours la surveillance, entendant un bruit de chute devrait être capable de vérifier la cause de ce bruit entendu pour s'assurer qu'aucun humain n'est en danger.

On souhaite donc s'intéresser à une méthode de classification de sons adaptée aux environnements intérieurs (maison, appartement) dans le cadre de la surveillance d'êtres humains : cela signifie prendre en compte toutes sortes de sons venant de l'environnement du robot et le rattacher à une classe permettant de guider les décisions du robot quant à ses prochains déplacements. Toutefois, un robot n'ayant pas les mêmes capacités d'analyse que l'être humain, deux problèmes principaux sont ainsi soulevés :

- **Quelles classes de sons rechercher ?** L'ensemble des classes proposées ne peut pas se résumer simplement à "Humain" et "Tout le reste". En effet, les sons marqueurs de la présence d'un être humain peuvent être très différents (figure 2.1)
- **Quelle méthode de classification choisir ?** Les méthodes de classification existantes sont naturellement moins performantes qu'un être humain. Or dans le cas d'un robot, une erreur peut s'avérer fâcheuse (par exemple, classifier un bruit de chute comme de la musique pourrait être dangereux). Il est donc nécessaire de prendre en compte cette possibilité d'erreur dans la classification.

Pour répondre à ces problèmes, nous proposons (i) l'utilisation d'un petit nombre de classes permettant de faire face à une grande variabilité des sons existants en intérieur, et (ii) l'utilisation d'une méthode de classification basée sur les fonctions de croyance permettant l'utilisation d'une classe de doutes lorsque la classification d'un son est trop difficile : en effet, on rappelle ici qu'on considère le son comme une modalité parmi d'autres, dans le cadre d'un système robotique de surveillance. Considérant ce facteur, et la variabilité des sons audibles au sein d'un environnement intérieur, une classification laissant la place au doute semble être adaptée. Ce système de classification a fait l'objet d'une publication [Lab+15].

Pour la suite de ce chapitre nous proposons un bref état de l'art en section 2.2. La section 2.3 décrit les différents éléments de la méthode proposée pour la classification : taxonomie,



(a) Parole (b) Musique

(c) Bruit de pas (d) Claquement de porte

FIGURE 2.1 – Quelques exemples de signaux sonores : en abscisse, le temps, et en ordonnée l'amplitude du signal.

méthode de classification et caractéristiques utilisées. Finalement la section 2.4 montre les résultats de la classification proposée sur une base de données que nous avons construite.

2.2 État de l'art sur la classification d'événements sonores

2.2.1 Définition du problème de classification

Le problème de la classification de sons peut être décrit de la manière suivante : on considère un signal acoustique (ou son) S, échantillonné à une fréquence F_e, dont les échantillons sont dénommés S(i) pour i ∈ [1,N] (figure 2.2), $\frac{N}{F_e}$ étant la durée du signal en secondes. On souhaite déterminer la classe de ce son de nature inconnue, parmi l'ensemble des classes $\mathcal{C} = \{C_1 \cdots C_J\}$ connues. Pour cela, on dispose d'une base de sons déjà étiquetée $\mathcal{T} = \{T_i, c_i\}_{i=1}^{M}$, où pour tout i, T_i est un son et $c_i \in \mathcal{C}$ est la classe qui lui est associée.

Ce problème est donc celui d'une classification classique : comment à partir de données connues, utiliser ce savoir et l'étendre pour classifier des données inconnues ?

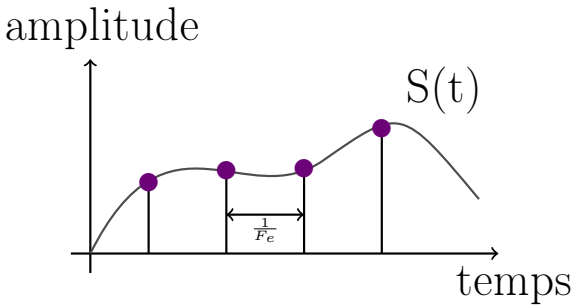


FIGURE 2.2 – Exemple d'un signal S(t) échantillonné

Le domaine de la classification de sons a été largement exploré dans la littérature, dans différents cadres d'application, et prend ses sources et ses techniques dans le domaine de la reconnaissance automatique de parole [TC02]. La structure des systèmes de classification de sons est souvent la même : une première étape d'extraction de caractéristiques du son a lieu, puis un algorithme de classification, issu la plupart du temps de la littérature de l'apprentissage automatique (machine learning), est appliqué.
Ainsi un système de classification est caractérisé par trois éléments principaux :

- **La taxonomie**, soit l'organisation des classes. Elle dépend principalement du but de la classification, c'est à dire de l'application à laquelle se destine le système.
- **Les caractéristiques extraites**, soit l'espace de description utilisé. Chaque son est représenté dans l'espace créé par ses caractéristiques et analysé dans cet espace. Le but est donc de déterminer quelles sont les caractéristiques répartissant "naturellement" les sons entre les différentes classes de la taxonomie utilisée. Il existe de nombreuses caractéristiques pour la classification audio dans la littérature. Certaines de ces caractéristiques seront décrites et étudiées en section 2.4.2.
- **La méthode de classification :** ce choix ne dépend pas nécessairement des caractéristiques utilisées, ou de la taxonomie, mais plutôt de la manière dont on cherche à classifier. De nombreux algorithmes de classification proposent pour chacun des éléments à classifier une unique classe d'appartenance : on se référera à ce genre de méthodes comme étant de la classification "dure". C'est par exemple le cas de la méthode des K plus proches voisins (KNN) [CH67], ou de la machine à vecteurs de support (SVM)[SS01]. D'autres algorithmes, en revanche, proposent des degrés d'appartenance pour chacune des classes : on se réfèrera à ce genre de méthodes comme étant de la classification "souple". C'est le cas de la méthode du C-means flou [Dun73].

De nombreux travaux en classification ont pour point commun les caractéristiques utilisées. Comme celles-ci seront étudiées plus en détails en section 2.4.2, nous proposons une présentation des principaux travaux pertinents effectués en classification audio, sans entrer dans le détail du calcul de ces caractéristiques.

2.2.2 Présentation des travaux existants

Dans les années 90, avec l'arrivée de nouveaux médias tels que le web, un besoin de nouvelles méthodes de recherche d'information (information retrieval) se fait sentir. Beaucoup de travaux étant déjà été effectués sur l'image et les vidéos, on voit apparaître des recherches sur la classification audio, comme [Li00], où la problématique est la suivante : considérant un son dit "requête" (query) et une base de sons appris appelés "prototypes", le système de recherche renvoie une liste des prototypes correspondant le mieux à la requête. Dans son travail, [Li00] souligne la performance de certaines caractéristiques, appelées coefficients cepstraux (MFCC), pour la classification.

2.2.2.1 Séparation parole-musique

Dans le même temps, des travaux sur la discrimination entre musique et parole commencent à apparaître, aidés par une nouvelle facilité d'accès à ce type de sons, grâce à internet. [Sau96] présente un travail principalement basé sur une caractéristique simple (le taux de passage par zéro du signal) et des statistiques qui lui sont liées. Il affiche d'excellents résultats de discrimination entre parole et musique dans un flux audio en temps réel, à l'aide d'un classifieur gaussien multivarié.

[Foo97] utilise les MFCC pour le même type de discrimination. Cependant l'algorithme d'apprentissage est basé sur l'utilisation d'arbres de quantification : un arbre est construit à partir des données d'apprentissage cherchant à diviser l'espace des caractéristiques en zones maximisant l'information mutuelle entre les données et leur étiquette. Cela revient à calculer un nombre fixe de seuils cherchant à séparer linéairement au mieux les différentes classes.

Ces travaux proposent une taxonomie simple (deux classes : musique ou parole), avec une classification dure : une décision sur la classe est prise sans tenir compte d'une incertitude sur la décision. Par la suite des taxonomies un peu plus complexes commencent à faire leur apparition : [TC02] propose une différenciation bien plus fine de différents genres musicaux (figure 2.3).

[TC02] se concentre principalement sur l'analyse des performances de 3 ensembles de caractéristiques. Les algorithmes de classification testés sont des mélanges de modèles gaussiens et des K plus proches voisins (KNN). Les performances affichées sont proches des capacités de classification humaines (un humain étant lui aussi sujet à erreur pour classifier différent types de musique). La taxonomie, très fournie, ne sort pas du cadre musique/parole, et semble donc insuffisante pour répondre au problème que l'on se pose ici.

2.2.2.2 Introduction de sons de l'environnement dans la taxonomie

[LZJ02] propose une taxonomie où apparaît la notion de "bruit d'environnement" (figure 2.4). Les règles de classifications, basées sur des seuillages adaptatifs sont simples, et la classi-

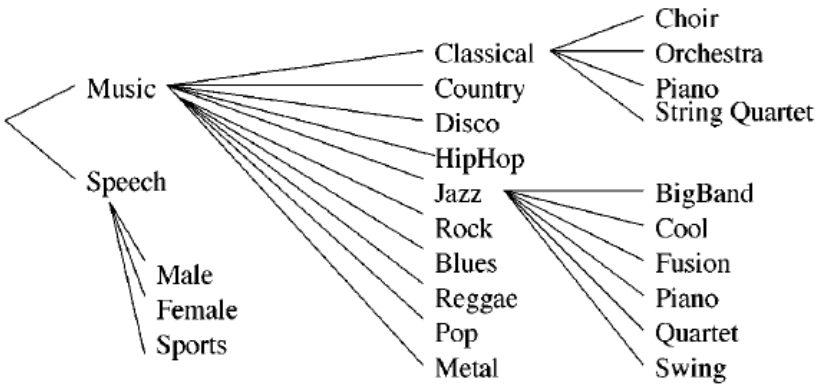


FIGURE 2.3 – Taxonomie proposée par [TC02]

fication a lieu en deux temps (tout d'abord une discrimination entre parole et les autres types de son, puis entre musique, environnement et silence.)

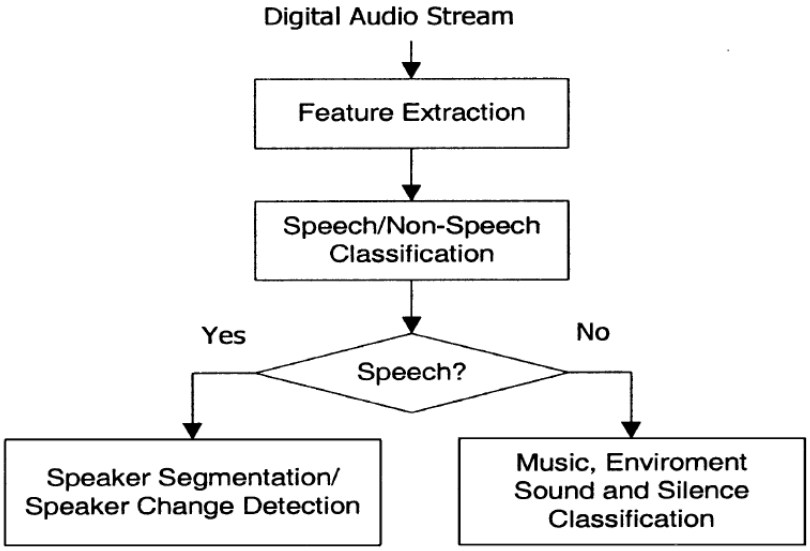


FIGURE 2.4 – Taxonomie proposée par [LZJ02]

Il est intéressant de noter que dans ce cas, le classifieur est le moins performant pour la catégorie "sons d'environnement" et reporte principalement ses erreurs sur la classe parole.

On notera aussi que la majorité des travaux produisent une classification dure, sans se soucier de "degré d'appartenance" à une classe. Chaque son est catégorisé sans prendre en compte l'incertitude liée à l'algorithme de classification ou aux caractéristiques utilisées. Il existe cependant des travaux utilisant des méthodes de classification plus souples : [HK13] décrit une classification basée sur un algorithme C-means souple. La taxonomie proposée, qui mélange différent types de sons (parole avec musique, parole avec bruit), semble bien se prêter

à l'utilisation d'algorithmes mesurant le degré d'appartenance. Cependant ce degré n'est pas utilisé, et c'est la classe ayant le plus haut degré d'appartenance qui est considérée comme celle du son analysé.

2.2.2.3 Autres taxonomies

Lorsqu'on s'éloigne du domaine de la discrimination entre parole et musique, il existe principalement deux types de travaux :

— Les travaux proposant une taxonomie de type "1 contre tous", c'est à dire de la classification à 1 classe (plus une classe de rejet). [IY08] cherche par exemple à classifier des bruits de pas à l'aide de caractéristiques basées sur le cepstre, tandis que [ZLG09] travaille sur la détection de chute dans un flux audio. L'avantage de ce genre de travaux est leur spécialisation : ils ne nécessitent de connaître que les caractéristiques d'une seule classe. Cette spécificité a pour coût le cadre d'application restreint.

— Les travaux proposant une taxonomie composée d'un grand nombre de classes. Un exemple est le travail réalisé par [SJB14 ; SB15] sur la classification de sons en environnement urbain. La taxonomie proposée est montrée en figure 2.5. Cette taxonomie de très grande taille nécessite l'apprentissage d'un grand nombre de classes très précises (il existe par exemple une classe taxi et une classe bus). Cela rend la création d'une base de données pour apprentissage beaucoup plus compliquée. Plus récemment [Pic15 ; SB16] ont utilisé la base de données présentée par [SJB14] pour classifier les sons à l'aide de réseaux de neurones convolutifs. Cet algorithme, adapté à des bases de données de grandes tailles, est très performant sur la base (76% de précision).

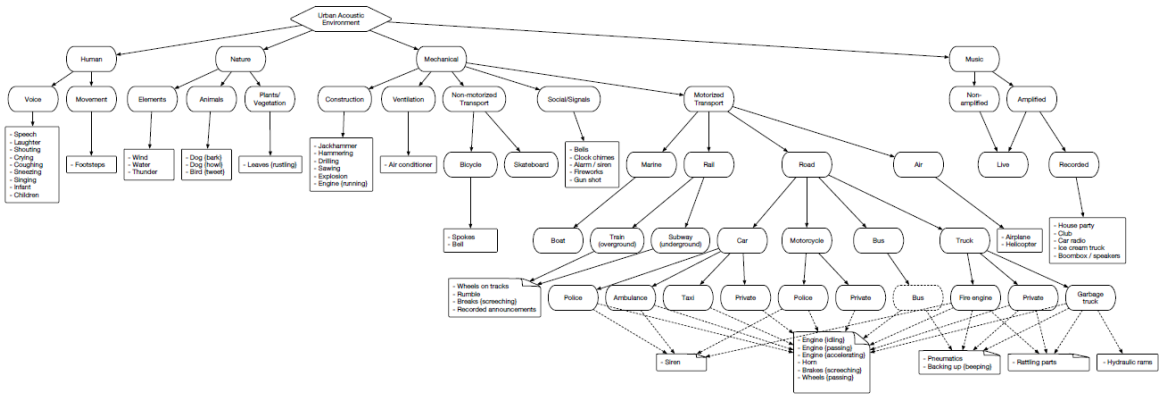


FIGURE 2.5 – Taxonomie proposée par [SJB14]

Peu de travaux existent sur la classification audio appliquée à la robotique de services, les travaux existants se concentrant plutôt sur la localisation de la source sonore [VSC15 ; NC15]. On notera une application proposée par [Jan+12], sur une classification de sons en environnement intérieur sur un robot immobile. La taxonomie proposée est divisée en 4 grandes catégories (tableau 2.1) : Prosodie, Cuisine, Mouvement, et Alarmes. Cependant les classes sont très précises (Alarme de four à micro-ondes, alarme de réfrigérateur), et apprises dans un

environnement unique où le robot a enregistré du son pendant un long moment, ce qui rend l'adaptabilité du système à tous types d'environnement intérieur difficile.

Prosodic	**Cooking**	**Moving**	**Alarms**
Eating	Cutlery	Open/Close a drawer	Microwave
Choking	Fill a glass	Move a chair	Fridge
	Running the tape	Open the microwave	Toaster
		Close the microwave	

TABLE 2.1 – Taxonomie proposée par [Jan+12]

2.2.3 Synthèse de l'état de l'art

Il n'existe pas à notre connaissance de travaux de classification de sons destinés aux robots compagnons. De nombreux moteurs de reconnaissance de parole existent et sont souvent mis à disposition en robotique, mais une classification dans un contexte plus général n'existe pas encore. Il est donc nécessaire de définir une taxonomie adaptée.

Par ailleurs, la quantité d'algorithmes de classification utilisés par les méthodes de la littérature mettent clairement en lumière le fait qu'il n'existe pas une méthode de classification supérieure à toutes les autres. Le choix d'une méthode adaptée au type de classification souhaité reste donc ouvert.

On note tout de même l'absence d'utilisation de classification souple. Or on se place ici dans un cadre multimodal : le robot est équipé de plusieurs capteurs lui donnant des informations partielles sur l'environnement qui l'entoure. C'est dans cette situation qu'une classification souple, avec degré d'appartenance du son entendu aux différentes classes peut être utile, et permettre au robot de prendre une décision pondérée par l'incertitude de sa détection.

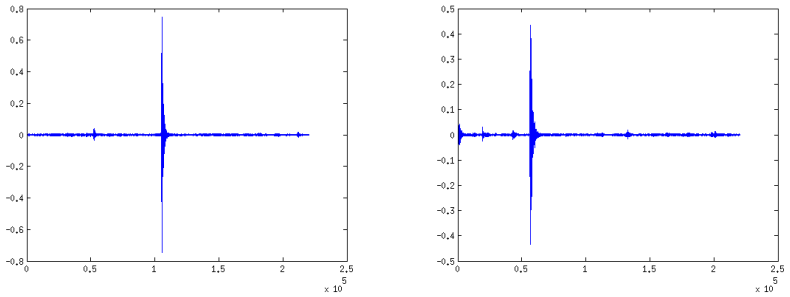
2.3 Classification d'événements sonores en environnement intérieur à l'aide de fonctions de croyance

L'état de l'art présenté dans la section 2.2 décrit principalement les taxonomies et les méthodes de classification utilisées dans la littérature. On détaille dans cette partie les choix faits dans notre cas pour les trois éléments principaux d'un système de classification : la taxonomie (section 2.3.1), la méthode de classification (section 2.3.2) et les caractéristiques étudiées (section 2.3.3). La méthode de classification utilisée étant tirée de la théorie crédibiliste, on propose donc en section 2.3.2 une description des bases des fonctions de croyance, suivie de la description détaillée de la méthode, illustrée par un exemple concret.

2.3.1 Taxonomie choisie

Comme indiqué précédemment, nous voulons classifier des sons acquis en environnement intérieur dans la vie courante d'êtres humains. Or, les marqueurs sonores de la présence d'êtres humains sont nombreux, ce qui rend la définition d'une taxonomie délicate. Par exemple [Jan+12] s'était concentré sur une taxonomie précise limitant le nombre de sons à classifier. L'environnement sonore d'un appartement est caractérisé par deux facteurs qui rendent difficile la classification :

— Sa variété : il existe une grande quantité de sons audibles au sein d'un environnement composé d'êtres humains. Chaque mouvement ou action effectué par des humains produit du signal sonore, souvent de manière très différente : claquer une porte, bruits de pas, bruits de vaisselles, froisser un papier, taper à l'ordinateur... tous ces signaux ont des formes et des caractéristiques variées. Si on souhaitait extraire le contenu sémantique du son entendu à tout moment à la manière d'un humain, la taxonomie nécessaire serait de très grande taille comme [SB15] (différence entre bus, voiture, taxi, et benne à ordures ménagères), ou il faudrait se concentrer sur un ensemble de sons précis comme le fait [Jan+12] (différence entre ouvrir et fermer un four à micro-ondes, mais pas de détection de parole). La taxonomie proposée par [Jan+12] serait plus adaptée à la détection d'évènements précis, où pour chaque évènement, une réaction adaptée du système est mise en place (par exemple, si l'alarme du four à micro-ondes est détectée, et qu'après un certain temps la porte du four n'a pas été ouverte, le robot pourrait émettre une petite alarme signifiant à l'humain qu'il peut aller chercher sa nourriture). La recherche d'exhaustivité dans la taxonomie proposée par [SJB14] est elle aussi exclue dans notre cas : une taxonomie de grande taille nécessite une base d'apprentissage adéquate pour chacune des catégories de sons considérées (ce qui n'est aujourd'hui pas disponible pour un environnement intérieur), mais augmente aussi la dimension du problème. Qui plus est, dans une taxonomie de grande taille, des classes sémantiquement éloignées peuvent être très proches en terme de contenu : c'est le cas par exemple donné en figure 2.6. Dans un cas, un signal contenant un claquement de fenêtre, et dans l'autre un signal contenant la chute d'un objet sur une table. Il apparaît que dans le cas d'un robot, où cette classification peut servir de guide aux prochaines actions, classifier un claquement de porte comme étant une chute peut poser problème dans la décision du robot.
— Sa variabilité : un son appartenant à la même classe sémantique peut être réalisé de plusieurs manières : considérons par exemple la classe "ouvrir un tiroir" proposée par [Jan+12]. Cette classe en elle-même contient une variabilité forte, et certaines réalisations de l'ouverture peuvent être très différentes les unes des autres. Cela provient à nouveau de la précision sémantique des classes. Une classe précise sémantiquement parlant n'est pas forcément séparable en ce qui concerne ses signaux réels. Ici encore, ce genre de classe est adapté si on souhaite apprendre un environnement précis, c'est à dire entraîner le robot à chaque fois qu'il est mis en place dans un endroit. Nous souhaitons obtenir une méthode de classification adaptable, qui permettrait de guider facilement le robot.



(a) Impact d'un objet sur une table

(b) Fermeture d'une fenêtre

FIGURE 2.6 – Un exemple de sons sémantiquement éloignés, mais de formes similaires.

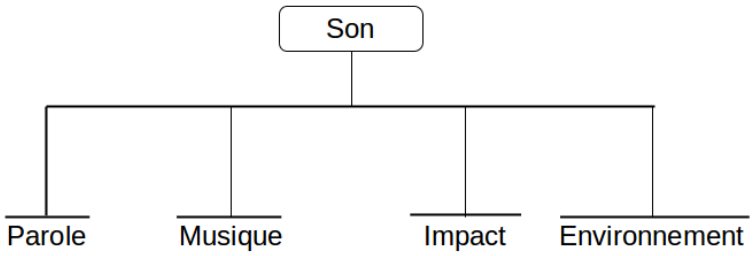


FIGURE 2.7 – La taxonomie proposée pour le présent système de classification de sons

Il apparaît ainsi que la taille de la taxonomie (le nombre de classes) et le contenu sémantique (la signification) des classes qui la composent sont les deux éléments principaux qui permettent de guider le choix de la taxonomie utilisée. Une trop grande taxonomie (donc avec un grand nombre de classes), semble inadapté : en effet, un grand nombre de classes implique de devoir apprendre sur des plus grandes bases de données, et est souvent lié à une sémantique précise. De même, des classes sémantiquement trop précises peuvent pousser le robot à l'erreur lors de la classification dans le cas où deux classes très différentes sémantiquement sont proches en ce qui concerne les caractéristiques. Or, on cherche à classifier les marqueurs principaux de la présence d'êtres humains en environnement intérieur. Prenant ces éléments en compte, nous proposons la taxonomie suivante (figure 2.7) :

— **Parole** : C'est le premier marqueur de la présence d'humains dans la scène, et la manière principale de l'humain de communiquer avec un robot. La parole sera ainsi le premier guide pour le robot. C'est aussi le cas pour un humain : la voix est un des meilleurs attracteurs d'attention pour l'être humain en général [SPG13].
— **Musique** : La musique est un type de son très souvent entendu en environnement intérieur : avec la présence de médias tels que la télévision, la radio ou les chaînes hifi, les possibilités sont multipliées. La musique risque d'être un indicateur pour un robot que l'information sonore n'est pas pertinente, car elle ne mènera pas à l'humain (mais

plutôt au moyen de diffuser de la musique). Cependant cette information peut être utile dans le cadre où la configuration de l'environnement du robot est connue : par exemple, si le robot entend de la musique provenant de la direction où il "sait" que se trouve la télévision, il pourra aller explorer les fauteuils placés devant la télévision en priorité, en espérant y détecter des humains.

— **Impacts** : Les impacts sont tous les sons brefs de forte énergie. Nous faisons ici deux hypothèses : (i) ce genre de sons est difficile à différencier d'un type d'impact à un autre (figure 2.6), et (ii) les sons d'impacts sont importants pour attirer l'attention d'un robot et sont souvent caractéristiques d'une action liée à l'humain : claquement de portes signifiant l'entrée ou le départ d'un être humain dans la scène, bruits de chutes, objets entrechoqués, le nombre d'exemples disponibles dans la vie courante d'êtres humains est grand, comme on le verra en section 2.4.1. C'est pourquoi nous choisissons cette classe à la sémantique large, regroupant des sons dont l'origine pourra être analysée plus en détail par le robot à l'aide d'autres capteurs.

— **Sons d'environnement** : Les sons d'environnement sont définis ici comme les sons de longue durée provenant de l'environnement du robot, parole et musique exclues. Cette classe compose une bonne partie des sons entendus par le robot (bruits de frictions, bruits de pas, grincement, bruits de fonds type réfrigérateur/four, etc..). Ces sons ne sont pas toujours significatifs de la présence d'un être humain, et possèdent des caractéristiques variées.

Nous proposons donc une taxonomie de petite taille, dont les classes sont volontairement sémantiquement peu précises. Ceci a le désavantage de ne pas donner au robot la capacité d'identifier précisément la source du son entendu (on ne fera par exemple pas la différence entre la porte du salon et la fenêtre de la salle de bain). Cependant, une telle taxonomie présente des avantages : (i) elle est adaptable à d'autres environnements intérieurs, car elle ne dépend pas des caractéristiques dudit environnement, (ii) dans le cas où une taxonomie sémantiquement plus précise serait utilisée, notre taxonomie peut servir de pré-traitement, gérant le type d'évènements qu'on souhaite classifier plus en détail, (iii) elle permet de donner un poids à l'audio dans le cadre de la robotique. En effet, les traitements audio étant peu robustes à des conditions réelles (bruits, diversité, sons inconnus, etc..), une classification avec une taxonomie de petite taille permet au robot d'avoir une première idée du son entendu.

2.3.2 Méthode de classification

Pour classifier les sons, il est nécessaire de s'adapter à la taxonomie choisie, ainsi qu'au domaine d'application auquel prétend le système. On souhaite pouvoir faire face à l'impossibilité pour le robot de classifier certains sons. Les classes étant sémantiquement peu précises, il est possible que certains sons soient difficiles à classifier car pouvant être proches de plusieurs classes. Dans ce genre de situation, on souhaite être en mesure de modéliser l'ignorance liée à la classification. A la différence de beaucoup de systèmes, un son classifié comme inconnu peut être une information intéressante pour un robot. Cela peut signifier que s'il n'a rien de plus important à faire, il doit chercher à savoir quelle était la source de ce bruit. Le cadre évidentiel des fonctions de croyance permet de modéliser l'ignorance et privilégie le principe

de prudence dans la classification. Il semble donc adapté au problème posé ici. Cette section se divise en deux sous-parties : quelques éléments de bases sur les fonctions de croyance sont tout d'abord introduits, puis l'algorithme de classification utilisé, le belief-KNN est décrit en détail.

2.3.2.1 Quelques éléments sur les fonctions de croyance

La théorie des fonctions de croyance provient du formalisme de Dempster-Shafer [Sha76]. En 1994, [SK94], introduit le modèle des croyances transférables (TBM), un cadre formel pour la représentation et la combinaison des connaissances. Le TBM est basé sur la définition de fonctions de croyance fournies par des sources d'information pouvant être complémentaires, redondantes et éventuellement non-indépendantes. Il propose un ensemble d'opérateurs permettant de combiner ces fonctions. On présente ici des éléments de base du formalisme du TBM. Dans la suite de ce manuscrit, on se référera au formalisme des fonctions de croyance comme au formalisme évidentiel ou crédibiliste de manière interchangeable.

Distribution de masses

On considère un système dont on souhaite déterminer l'état réel. Le système peut prendre un nombre fini connu d'états. L'ensemble des états ou hypothèses s_i, i$\in [1, P]$ possibles que peut prendre le système est appelé espace de discernement et est classiquement noté Ω.

$$\Omega = \{s_1 \cdots s_P\} = \bigcup_{i=1}^{P} \{s_i\} \tag{2.1}$$

Le nombre d'hypothèses contenu dans Ω, soit $|\Omega|$, est appelé cardinal et les hypothèses de l'espace de discernement sont exclusives.

Une fonction de croyance sur Ω peut être représentée de plusieurs façons, la plus courante étant la **distribution de masses** (basic belief assignment en anglais, soit BBA). Une distribution de masse est une fonction notée m^Ω définie par :

$$\begin{array}{rccc} m^\Omega & : & 2^\Omega & \rightarrow & [0,1] \\ & & \mathrm{H} & \rightarrow & m^\Omega(H) \end{array} \tag{2.2}$$

où $2^\Omega = \{\emptyset, \{s_1\}, \{s_1, s_2\}, \{s_1, s_3\}, \cdots, \{s_1, \cdots, s_{P-1}\}, \Omega\}$ est l'ensemble des sous-ensembles de Ω, de cardinal $|2^\Omega| = 2^{|\Omega|} + 1$. Une source de connaissance l produisant de l'information sur 2^Ω peut ainsi produire une distribution de masses, notée m_l^Ω. La valeur $m_l^\Omega(\mathrm{H})$ attribuée à $\mathrm{H} \in 2^\Omega$ est la croyance qui supporte l'hypothèse H.

Toute distribution de masses respecte la règle suivante :

$$\sum_{H \subset 2^\Omega} m^\Omega(H) = 1 \tag{2.3}$$

Une distribution de masses ayant toute la croyance concentrée sur Ω est appelée distribution de masses vide, tandis qu'une distribution de masses contenant l'intégralité de la croyance sur une seule hypothèse (autre que Ω et $\emptyset$) est appelée distribution de masse catégorique. Les ensembles d'une distribution de masses dont la masse est non-nulle sont appelés les éléments focaux de cette distribution de masses. Dans la suite de ce manuscrit, les termes "distribution de masses" et BBA seront utilisés de manière interchangeable.

Exemple : Un robot se trouve en situation d'interaction avec une personne. Pour pouvoir ajuster son comportement, il dispose d'un classifieur souple permettant de déterminer si la personne à laquelle il a affaire est un adolescent, un adulte, ou une personne âgée. Le classifieur lui donne comme réponse 10% de chance d'appartenance à la classe "adolescent". La personne en présence ayant les traits d'un adulte d'âge mûr, le classifieur est à 70% capable de savoir si la personne est soit un adulte, soit une personne âgée. Les 20 % restants concernent le doute quant à la capacité du classifieur à associer la personne à son âge. La distribution de masses de croyance associée au type de personne se trouvant face au robot, définie sur l'espace de discernement $\Omega = \{Ado, Adu, PA\}$ (pour signifier respectivement "adolescent", "adulte", et "personne âgée") est résumée dans le tableau 2.2

	{Ado}	{Adu}	{PA}	{Ado,Adu}	{Ado,PA}	{Adu,PA}	Ω	$\emptyset$
m^Ω	0.1	0	0	0.7	0	0	0.2	0

TABLE 2.2 – Exemple de distribution de masses

Fusion de distribution de masses

Il existe des opérateurs pour fusionner des fonctions de croyance, lorsque plusieurs sources de connaissances sont disponibles sur le même espace de discernement [Den08]. Un des opérateurs les plus utilisés est la fusion conjonctive, notée $\textcircled{\cap}$. Elle est définie pour deux BBA m_1^Ω et m_2^Ω, pour tout H,A,B $\subset \Omega$ par :

$$m_{1\textcircled{\cap}2}^\Omega(H) = (m_1^\Omega \textcircled{\cap} m_2^\Omega)(H) = \sum_{A \cap B = H} (m_1(A) \cdot m_2(B)) \tag{2.4}$$

La combinaison conjonctive concentre donc la croyance de la BBA combinée sur l'intersection des ensembles des BBA sources. C'est un opérateur de combinaison de spécialisation. Il existe des cas où les sources produisent des connaissances contradictoires, c'est à dire sur des ensembles de 2^Ω disjoints : dans ce cas la masse résultante est transférée sur $\emptyset$, ce qui peut être génant. On parle dans ce cas de conflit. $\emptyset$ est un ensemble absorbant (comme son intersection avec n'importe quel autre ensemble est $\emptyset$, la masse combinée sera transférée à $\emptyset$ lors de combinaisons), et si les sources sont combinées plusieurs fois, l'ensemble vide gagnera artificiellement de la croyance. Attribuer de la croyance à $\emptyset$ est mathématiquement juste, mais a peu de sens pour l'interprétation : comment comprendre une croyance sur l'espace vide ?

Dans la majorité des applications utilisant les fonctions de croyance, le conflit est géré à chaque combinaison. Dempster [Dem08] propose une normalisation pour gérer le conflit. Ceci

a donné la règle orthogonale de Dempster notée $\oplus$ et définie, pour tout H,A,B $\subset 2^{\Omega}$ par :

$$m^{\Omega}_{1\oplus 2}(H) = (m_1 \oplus m_2)(H) = \begin{cases} \frac{m^{\Omega}_{1\bigcirc 2}(H)}{1-m^{\Omega}_{1\bigcirc 2}(\emptyset)} & \forall H \subset \Omega \quad \text{si } H \neq \emptyset \\ 0 & \text{si } H = \emptyset \end{cases} \tag{2.5}$$

Cette normalisation ne prend pas en compte l'origine du conflit, mais permet d'obtenir une représentation du savoir interprétable, où la croyance attribuée à l'ensemble vide est nulle.

Exemple : On considère maintenant que le robot dispose de deux classifieurs différents l'informant sur son type d'interlocuteur, donc deux BBA définie sur l'espace de discernement de l'exemple précédent : $\Omega = \{Ado, Adu, PA\}$. Ces deux BBA, m^{Ω}_1 et m^{Ω}_2 sont définies dans le tableau 2.3.

	{Ado}	{Adu}	{PA}	{Ado,Adu}	{Ado,PA}	{Adu,PA}	Ω	$\emptyset$
m^{Ω}_1	0.1	0	0	0.7	0	0	0.2	0
m^{Ω}_2	0.1	0.3	0	0	0.2	0	0.4	0

Table 2.3 – Définition d'un couple de BBA qu'on souhaiterait combiner

Pour combiner ces deux BBA, il est nécessaire de calculer les intersections des ensembles focaux de chacune des BBA (tableau 2.4).

	{Ado}	{Adu}	{PA}	{Ado,Adu}	{Ado,PA}	{Adu,PA}	Ω	$\emptyset$
{Ado}	{Ado}	$\emptyset$	$\emptyset$	{Ado}	{Ado}	$\emptyset$	{Ado}	$\emptyset$
{Adu}	$\emptyset$	{Adu}	$\emptyset$	{Adu}	$\emptyset$	{Adu}	{Adu}	$\emptyset$
{PA}	$\emptyset$	$\emptyset$	{PA}	$\emptyset$	{PA}	{PA}	{PA}	$\emptyset$
{Ado, Adu}	{Ado}	{Adu}	$\emptyset$	{Ado,Adu}	{Ado}	{Adu}	{Ado,Adu}	$\emptyset$
{Ado,PA}	{Ado}	$\emptyset$	{PA}	{Ado}	{Ado,PA}	{PA}	{Ado,PA}	$\emptyset$
{Adu,PA}	$\emptyset$	{Adu}	{PA}	{Adu}	{PA}	{Adu,PA}	{Adu,PA}	$\emptyset$
Ω	{Ado}	{Adu}	{PA}	{Ado,Adu}	{Ado,PA}	{Adu,PA}	Ω	$\emptyset$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Table 2.4 – Intersection des différents éléments de 2^{Ω}

En se référant au tableau 2.4, il apparaît que la combinaison conjonctive aura 6 éléments focaux ({Ado}, {Adu}, {Ado,Adu}, {Ado,Pa}, Ω, et $\emptyset$), tandis que la combinaison de Dempster en contiendra 5 (les mêmes que pour la combinaison conjonctive, moins $\emptyset$). Les éléments focaux de la combinaison conjonctive sont ainsi calculés de la manière suivante :

$$
\begin{aligned}
m^{\Omega}_{1\bigcirc 2}(\{Ado\}) =& m^{\Omega}_1(\{Ado\}) \cdot \left(m^{\Omega}_2(\{Ado\}) + m^{\Omega}_2(\{Ado, PA\}) + m^{\Omega}_2(\Omega)\right) \\
&+ m^{\Omega}_2(\{Ado\}) \cdot \left(m^{\Omega}_1(\{Ado, Adu\}) + m^{\Omega}_1(\Omega)\right) \\
&+ m^{\Omega}_1(\{Ado, Adu\}) \cdot m^{\Omega}_2(\{Ado, PA\})
\end{aligned} \tag{2.6}
$$

$$
m^{\Omega}_{1\bigcirc 2}(\{Adu\}) = m^{\Omega}_2(\{Adu\}) \cdot \left(m^{\Omega}_1(\{Ado, Adu\} + m^{\Omega}_1(\Omega)\right) \tag{2.7}
$$

$$
m^{\Omega}_{1\bigcirc 2}(\{Ado, Adu\}) = m^{\Omega}_1(\{Ado, Adu\}) \cdot m^{\Omega}_2(\Omega) \tag{2.8}
$$

$$
m^{\Omega}_{1\bigcirc 2}(\{Ado, PA\}) = m^{\Omega}_2(\{Ado, PA\}) \cdot m^{\Omega}_1(\Omega) \tag{2.9}
$$

$$
m^{\Omega}_{1\bigcirc 2}(\Omega) = m^{\Omega}_1(\Omega) \cdot m^{\Omega}_2(\Omega) \tag{2.10}
$$

$$
m^{\Omega}_{1\bigcirc 2}(\emptyset) = m^{\Omega}_1(\{Ado\}) \cdot m^{\Omega}_2(\{Adu\}) \tag{2.11}
$$

Et les masses résultantes, peuvent ainsi être trouvées dans le tableau 2.5

	{Ado}	{Adu}	{PA}	{Ado,Adu}	{Ado,PA}	{Adu,PA}	Ω	$\emptyset$
$m^{\Omega}_{1\bigcirc 2}$	0.3	0.27	0	0.28	0.04	0	0.08	0.03
$m^{\Omega}_{1\oplus 2}$	0.31	0.28	0	0.29	0.04	0	0.08	0

TABLE 2.5 – Résultats de la combinaison conjonctive et de Dempster

2.3.2.2 Présentation du belief-KNN

L'algorithme des K plus proches voisins, ou KNN [CH67], est un algorithme de classification classique très utilisé en apprentissage automatique. Sa version la plus simple est illustrée en figure 2.8. Le concept est le suivant : pour déterminer la classe d'un élément inconnu, on observe ses K plus proches voisins appartenant à la base de sons étiquetée $\mathcal{T}$ définie en section 2.2 dans l'espace de représentation choisi (l'espace de caractéristiques), au sens d'une distance pré-définie. La majorité du temps, dans des applications directes, la distance utilisée est la distance euclidienne. C'est cette distance qu'on utilisera dans la suite de ce manuscrit.

Pour prendre une décision sur la classe de l'élément, il existe plusieurs méthodes, la plus simple étant le vote majoritaire : on décompte le nombre d'éléments de chaque classe parmi les K plus proches voisins, et on attribue à l'élément inconnu la classe majoritaire. Par exemple, dans le cas de la figure 2.8, on essaie de classifier un élément inconnu (le rond noir) dans un espace de représentation de dimension 2. Si on choisit K=3, on voit que le vote majoritaire attribue la classe 1 à l'élément à classifier (2 contre 1 en faveur de la classe 1), tandis que si on choisit K = 5, c'est la classe 2 qui l'emporte à 3 contre 2. Ainsi, dans certains cas la classification d'un élément peut dépendre du nombre de voisins considéré.

Le KNN à vote majoritaire est la manière la plus simple de classifier un élément : de nombreux travaux ont été effectués avec cette méthode de classification et sur la manière de prendre une décision sur la classe, par exemple en pondérant les votes par la distance des voisins à l'élément à classifier, où en considérant un voisinage à double-sens [TH15]. L'algorithme

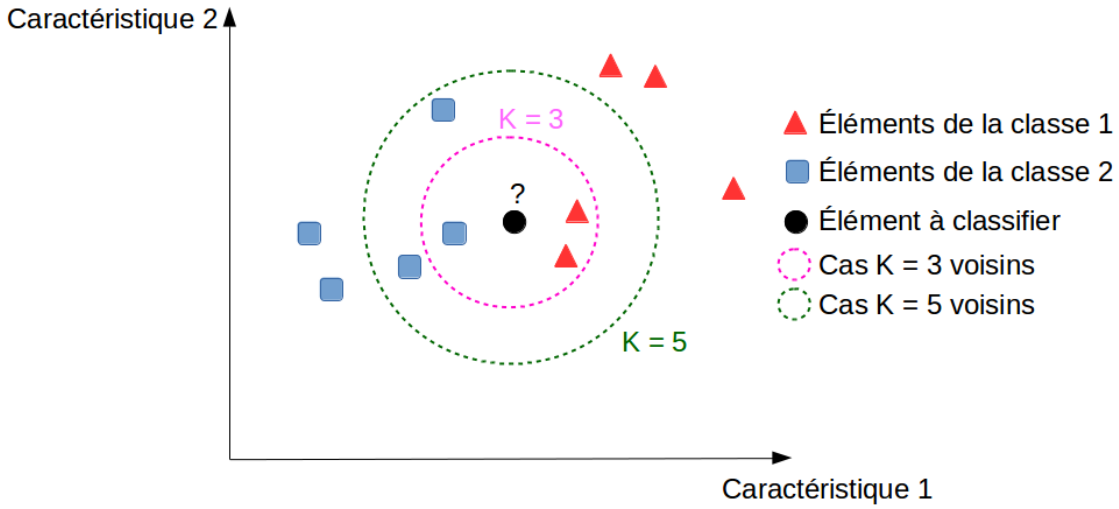


FIGURE 2.8 – Illustration de l'algorithme du KNN : lorsqu'on choisit K = 3, le vote majoritaire donne l'élement à classifier comme appartenant à la classe 1. Lorsqu'on choisit K = 5, le vote majoritaire donne l'élément à classifier comme appartenant à classe 2.

du KNN est soumis au fléau de la dimension : plus la dimension de l'espace de représentation augmente, plus le nombre de données d'apprentissage nécessaire croît. Plus la base d'apprentissage est grande, plus le temps de calcul pour trouver les plus proches voisins est grand. Il est donc judicieux de s'attacher à représenter les données à classifier dans un espace de la plus petite dimension possible.

Le belief-KNN est introduit par [Den95]. Le principe général de l'algorithme est le même que celui du KNN classique : on observe les plus proches voisins de l'élément à classifier, mais on cherche cette fois-ci à construire une BBA représentant la croyance en l'appartenance de l'élément à chacune des classes. L'espace de discernement est donc $\Omega = \mathcal{C} = \{C_1, \cdots, C_J\}$. L'algorithme se divise en trois étapes successives.

Tout d'abord, une BBA est attribuée à chacun des K voisins considérés, dépendant de sa classe. Pour le i^{eme} voisin n_i de la classe C_q, la BBA est construite de la manière suivante :

$$m_i^{\Omega}(\{C_q\}) = \alpha e^{-\gamma_q d^i} \tag{2.12}$$

$$m_i^{\Omega}(\Omega) = 1 - \alpha e^{-\gamma_q d^i} \tag{2.13}$$

$$m_i^{\Omega}(C_j) = 0 \quad \forall C_q \neq C_j \subset \Omega \tag{2.14}$$

où d^i représente la distance du voisin à l'élément que l'on cherche à classifier et γ_q est la vitesse de décroissance de la croyance en la classe C_q avec la distance du voisin. α est un facteur que l'on fixe souvent proche de 1 ($\sim$0.95) et qui signifie que même si l'élément à classifier se trouve au même endroit que le voisin dans l'espace de représentation, il est impossible d'être complètement certain que l'élément à classifier appartient à la classe C_q.

Plus le voisin considéré sera éloigné de l'élément à classifier, plus la croyance de sa BBA sur Ω sera forte, et la croyance sur la classe de ce voisin sera faible. Inversement plus le voisin considéré sera proche, plus la croyance de sa BBA sur la classe sera forte et la croyance sur Ω sera faible.

Une fois ces BBA créées, elles sont combinées classe par classe, i.e. toutes les BBA des voisins de classe C_q sont agrégées grâce à une combinaison conjonctive pour créer une BBA de classe m_q^Ω :

$$m_q^\Omega(\{C_q\}) = 1 - \prod_{n_i \in C_q} \left(1 - \alpha \exp^{\gamma_q d^i}\right) \tag{2.15}$$

$$m_q^\Omega(\Omega) = \prod_{n_i \in C_q} \left(1 - \alpha \exp^{\gamma_q d^i}\right) \tag{2.16}$$

On notera ici qu'il est impossible de créer du conflit, les BBA qui sont combinées ayant toutes les mêmes éléments focaux (C_q et Ω). La BBA m_q est représentative de la "force" de la classe C_q, dépendant du nombre de voisins de cette classe et de leur distance à l'élément à classifier.

Finalement, toutes les BBA de classe sont combinées. Ici, du conflit va forcément apparaître. Pour faire face à ce problème, [Den95] propose l'utilisation d'une combinaison de Dempster :

$$m^\Omega(\{C_q\}) = \frac{m_q^\Omega(\{C_q\}) \prod_{r \neq q} m_r^\Omega(\Omega)}{\eta} \tag{2.17}$$

$$m^\Omega(\Omega) = \frac{\prod_{q=1}^{J} m_q^\Omega(\Omega)}{\eta} \tag{2.18}$$

où η est la constante de normalisation de la combinaison de Dempster :

$$\eta = \sum_{q=1}^{M} \left[\prod_{r \neq q} m_r^s(C) \right] + (1 - M) \prod_{q=1}^{M} m_q^s \tag{2.19}$$

On supprime ainsi la croyance portée sur l'ensemble vide, ce qui permet de conserver une représentation de la connaissance directement interprétable par un robot. Au terme de cette fusion, on obtient pour l'élément à classifer une BBA m^Ω, dont les éléments focaux sont les différentes classes $\{C_q\}$, q $\in [1, J]$, ainsi que Ω.

Exemple : On reprend le cas montré en figure 2.8 : on souhaite déterminer la classe d'un élément parmi deux classes possibles, soit $\Omega = \{C_1, C_2\}$. Pour ce faire, on dispose d'un espace

de représentation de dimension 2 (2 caractéristiques extraites). On considère premièrement le cas "3 voisins", soit K = 3. On considère que les 3 voisins se trouvent à la même distance euclidienne de 1 de l'échantillon à classifier. Les différentes BBA sont données dans le tableau 2.6.

	$\{C_1\}$	$\{C_2\}$	Ω	$\emptyset$
m_1^Ω	0.84	0	0.16	0
m_2^Ω	0	0.6	0.4	0
$\mathbf{m^\Omega}$	**0.68**	**0.19**	**0.13**	**0**

TABLE 2.6 – Cas "3 voisins" : BBA attribuées par classe, et BBA finale donnée par la combinaison de Dempster

La BBA attribuée à l'élément classifié donne 68% de la croyance à la classe C_1, de manière juste (2 éléments sur 3 y appartenant).
Si cette fois nous considérons 5 voisins, nous passons à 3 voisins appartenant à la classe 2, et toujours les mêmes 2 voisins appartenant à la classe 1. On considère ici que les deux nouveaux éléments de la classe 2 sont respectivement à une distance euclidienne de 3 et 4 de l'élément qu'on cherche à classifier. Les différentes BBA correspondantes à ce cas sont données dans le tableau 2.7

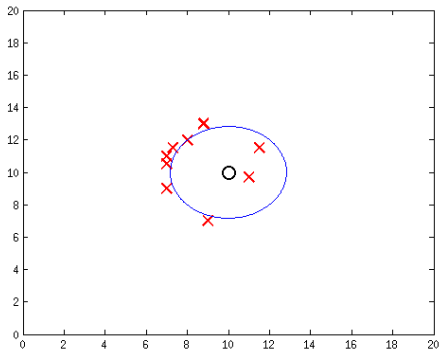
	$\{C_1\}$	$\{C_2\}$	Ω	$\emptyset$
m_1^Ω	0.84	0	0.16	0
m_2^Ω	0	0.76	0.24	0
$\mathbf{m^\Omega}$	**0.56**	**0.34**	**0.10**	**0**

TABLE 2.7 – Cas "5 voisins" : BBA attribuées par classe, et BBA finale donnée par la combinaison de Dempster

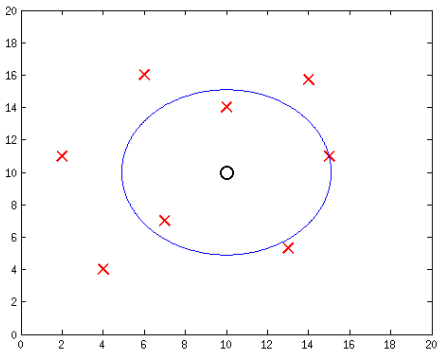
Le KNN à vote majoritaire donnerait cette fois-ci la classe C_2 gagnante. Le belief-KNN, lui, donne toujours une croyance plus forte sur la classe C_1. Ceci est dû à l'impact de la distance des différents voisins considérés à l'élément à classifier. Cet impact est fortement contrôlé par la valeur γ_q choisie pour chacune des classes C_q.

Sur l'influence de γ_q : Le paramètre γ_q a une influence majeure sur le comportement du belief-KNN et a été étudié dans la littérature [ZD98 ; LPD13]. Lors de l'attribution des BBA classe par classe, γ_q contrôle l'importance donnée à la distance des voisins de la classe q à l'échantillon considéré. Plus γ_q sera grand, moins les voisins lointains auront d'importance, et inversement. Chaque γ_q doit donc être fixé en fonction des données considérées. S'il existe une grande dispersion des données au sein d'une classe, il semble naturel de diminuer γ_q pour ne pas négliger d'éventuels voisins lointains et produire ainsi des masses contenant une croyance forte sur Ω. Inversement, si l'espace de description regroupe fortement les éléments de la classe q, on utilisera un γ_q plutôt petit pour ne pas donner trop d'influence à des voisins lointains lors de la classification.

[Den95] propose de fixer γ_q comme l'inverse de la distance moyenne des éléments de C_q deux à deux dans la base d'entraînement. Ceci permettra ainsi de prendre en compte la dispersion intra-classe dans l'attribution des masses de croyance. Un exemple est donné en figure 2.9.



(a) Cas d'une classe à faible dispersion (l'ellipse délimite les 3 plus proches voisins)



(b) Cas d'une classe à forte dispersion (l'ellipse délimite les 3 plus proches voisins)

FIGURE 2.9 – Fixer le paramètre γ_q pour la classe C_q : dans un cas (a) la classe est faiblement dispersée (FD), et un γ "grand" suffit (0.3), dans l'autre (b) la grande dispersion (GD) de la classe doit être prise en compte en diminuant la valeur de γ (0.12).

Dans le cas de la figure 2.9, on souhaite classifier un élément (cercle noir) dont tous les plus proches voisins appartiennent à la même classe (croix rouges). Pour ce faire, on utilise les 3 plus proches voisins. Deux cas sont présentés : la figure 2.9 (a) montre le cas dont les éléments sont faiblement dispersés (cas FD), tandis que la figure 2.9 (b) montre une classe plus dispersée (cas GD). Dans les deux cas, au vu des données, il semblerait logique d'obtenir une masse de croyance similaire. En effet, nous n'avons pas plus de doute en l'appartenance de l'élément à classifier dans le cas GD que dans le cas FD. Si on ne prend pas en compte la dynamique des données, et que l'on fixe le même γ_q (par exemple $\gamma_q = 0.4$) pour chacun des deux cas, on obtient des disparités fortes au sein des masses (tableau 2.8). Si on ajuste la valeur de γ_q comme le propose [Den95], cependant, on obtient des valeurs similaires dans le cas FD et GD (tableau 2.9). C'est cette valeur de γ_q ajustée que l'on utilisera dans la suite.

m_q^Ω	C_q	Ω
γ_q= 0.4, FD	0.82	0.18
γ_q= 0.4, GD	0.40	0.60

TABLE 2.8 – Masses sans ajustement avec γ_q fixe (0.4)

m_q^Ω	C_q	Ω
γ_q= 0.3, FD	0.89	0.11
γ_q= 0.12, GD	0.89	0.11

TABLE 2.9 – Masse avec ajustement comme proposé par [Den95]

2.3.3 Caractéristiques étudiées

Considérons un signal audio temporel S à analyser. La procédure d'extraction de caractéristiques est généralement la suivante : une fenêtre glissante W de taille N échantillons est appliquée avec recouvrement sur le signal. A chaque position de cette fenêtre, les caractéristiques sont extraites. Ces caractéristiques sont ensuite agrégées statistiquement pour avoir une représentation finale du signal.
Comme dit précédemment, de nombreuses caractéristiques sont disponibles pour la classification audio, et se retrouvent dans divers travaux. Elles se divisent généralement en trois sous-catégories : les caractéristiques temporelles, qui sont extraites directement du signal acquis ; les caractéristiques "spectrales", qui sont extraites de la transformée du signal dans le domaine de Fourier ; et les caractéristiques "transformées", qui sont issues de transformations plus complexes du signal. Voici les caractéristiques qui ont été étudiées dans le cadre de ce travail.

Zero-Crossing Rate (ZCR) : Comme son nom l'indique, le "zero-crossing rate" est une mesure temporelle de la fréquence de passage par zéro du signal. Il a été utilisé très tôt dans les travaux d'analyse de paroles [AR76] ainsi que dans des travaux de discrimination entre parole et musique. Il est défini de la manière suivante :

$$ZCR_t = \frac{1}{N}\sum_{k=1}^{N}\frac{1}{2}\left|sgn[S_t(k)] - sgn[S_t(k-1)]\right| \tag{2.20}$$

où $\mathrm{sgn}[\mathrm{S}_t(\mathrm{k})] = \begin{cases} 1, & S(k) \geq 0 \\ -1, & S(k) < 0 \end{cases}$ et $\mathrm{S}_t(\mathrm{k})$ est le k-ième échantillon du signal S au sein de la fenêtre W_t

La ZCR est généralement considérée comme une mesure du "niveau de bruit" du signal.

Moyenne quadratique de l'énergie (RMS énergie) et le pourcentage de trame à basse énergie (LE) : La moyenne quadratique de l'énergie au sein d'une fenêtre est une mesure temporelle directe de l'énergie du signal, elle est définie de la manière suivante :

$$RMSE_t = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\frac{S_t(k)}{max(|S_t|)}} \tag{2.21}$$

La RMSE peut être une indication sur la variation d'intensité au sein du signal, notamment au travers du pourcentage de trame à basse énergie :

$$LE = \frac{N_w}{100}\cdot \operatorname*{count}_t\left(RMSE_t < \frac{1}{N_w}\sum_{k=1}^{N_w} RMSE_k\right) \tag{2.22}$$

où N_w est le nombre total de fenêtres appliquées au signal S, et "count" est une fonction de décompte. Le pourcentage de trame à basse énergie est une mesure de la dispersion de l'énergie au sein du signal. Ainsi, un morceau de musique aura naturellement un pourcentage inférieur à un bruit de pas.

Flux spectral (SF) : Le flux spectral mesure les changements locaux dans le spectre.

$$SF_t = \sum_{k=1}^{N} \left(F_t(k) - F_{t-1}(k)\right)^2 \tag{2.23}$$

où $\mathrm{F}_t(\mathrm{k})$ est le k-ième coefficient de la transformée de Fourier de S_t. De hauts flux spectraux sont indicateurs de changement rapide au sein du spectre du signal. Le flux spectral est souvent employé pour différencier les styles musicaux, et peut être très utile pour discriminer certains sons en environnement intérieur.

Fréquence centrale (FC) et la largeur de bande (BW) : La fréquence centrale d'un signal est la mesure du centre de gravité fréquentiel du spectre.

$$FC_t = \frac{\sum_{k=1}^{N} k \cdot F_t(k)^2}{\sum_{k=1}^{N} F_t(k)^2} \tag{2.24}$$

La fréquence centrale ne permet pas de discriminer facilement les sons directement, deux sons très différents pouvant tout à fait avoir le même centre de gravité fréquentiel. Cependant elle permet de calculer une mesure beaucoup plus informative qu'est la largeur de bande :

$$BW_t = \sqrt{\frac{\sum_{k=0}^{N} \left(k - FC_t\right)^2 F_t(k)^2}{\sum_{k=0}^{N} F_t(k)^2}} \tag{2.25}$$

La largeur de bande est une mesure de l'étalement spectral du son. Par exemple, une symphonie aura un étalement spectral plus large qu'une personne qui parle.

Rolloff Spectral (SR) : Le rolloff spectral est défini comme étant la portion du spectre contenant un pourcentage prédéfini de l'énergie.

$$\sum_{k=1}^{R_t} F_t(k) = \alpha \sum_{k=1}^{N} F_t(k) \tag{2.26}$$

On définit généralement α assez grand (classiquement 0.9). Le rolloff permet de discriminer les sons dont l'énergie est répartie principalement dans les basses fréquences. C'est le cas par exemple des bruits de moteurs.

Coefficients cepstraux de Mel (MFCC) : Les coefficients cepstraux de Mel ont été employés massivement dans la reconnaissance de parole, la discrimination musique/parole, et continuent d'être utilisés dans toutes sortes d'applications en classification de sons.
Ces coefficients sont obtenus à partir de la transformée de Fourier du signal. Le spectre de puissance est ensuite pondéré par un banc de filtres triangulaires espacés selon l'échelle dite de "Mel", qui est une échelle liée à la perception humaine. La sortie de ces filtres est ensuite soumise à une transformée en cosinus discrète. Plus de détails peuvent être trouvés sur l'utilisation des MFCC dans [RJ93].

Il existe encore d'autres caractéristiques extraites de la littérature : entropie spectrale, spectre de puissance [HK13], LPCC (pour Linear Predictive Coding Coefficients) [IY08], périodicité de bande [LZJ02], mais celles décrites précédemment apparaissent le plus souvent dans la littérature. Nous avons donc décidé de limiter notre étude à ces caractéristiques.

En résumé, sur chacune des fenêtres glissantes sont extraites les caractéristiques suivantes : la zero-crossing rate (ZCR), la fréquence centrale (FC), la largeur de bande (BW), le roll-loff spectral (SR), le flux spectral (SF), et les 13 premiers coefficients de Mel (MFCC qui contiennent le plus d'information [TC02]), soit pour chaque fenêtre 18 caractéristiques.

Dans le but d'obtenir une représentation unifiée du signal sur l'intégralité de sa durée, on agrège statistiquement les caractéristiques acquises sur les fenêtres glissantes. Sans procèder ainsi, on obtient une représentation dépendante de la durée du signal (i.e. du nombre de fenêtres), dont l'information temporelle ne sera pas prise en compte lors de la classification, et dont les caractéristiques ne sont pas tout le temps représentatives de la nature du signal. De plus, la taille de la représentation peut devenir un problème : si on considère un signal de 10 secondes, sur lequel on fait glisser une fenêtre de taille 10ms avec un recouvrement de 50% (qui est le recouvrement standard), on obtiendrait ainsi une représentation de taille (2x100x10)x18 = 36000 pour un signal.

Pour obtenir la représentation finale, on utilise la moyenne et la variance de chacune des caractéristiques considérées sur l'intégralité des fenêtres ([TC02]). A cela vient s'ajouter le pourcentage de trame à basse énergie, calculé directement sur l'intégralité du signal (LE). Cela donne ainsi pour chaque signal une représentation de dimension fixe 37 (moyenne et variance des 18 caractéristiques déjà citées, plus le pourcentage de trames à basse énergie).

2.4 Résultats

2.4.1 Base de sons construite

Il existe peu de bases de données disponibles contenant des sons de la vie courante d'êtres humains. Les signaux de paroles et de musique sont facilement accessibles sur internet, mais des signaux représentatifs des 2 autres classes (impacts, sons d'environnement) ne sont pas

directement disponibles. Nous avons donc choisi de construire une base de données à partir de deux bases de données existantes :

— La base proposée par [TC02] : c'est de cette base qu'est extraite la majorité des signaux de paroles et de musiques de la base utilisée ici.
— [Vac+14] a crée un corpus multimodal dans lequel un être humain accomplit des activités de la vie courante (vaisselle, aspirateur, sieste, etc...) dans un appartement contenant 7 microphones répartis dans les différentes pièces et 4 caméras (figure 2.10) . M. Vacher et al. nous ayant gracieusement donné accès à ce corpus, nous avons pu extraire des événements correspondants aux éléments de la taxonomie proposée et les inclure dans la base de données utilisée ici.

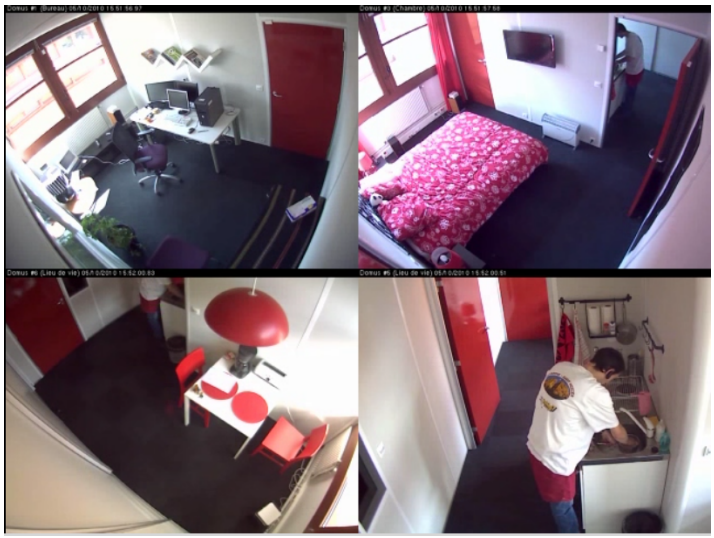


FIGURE 2.10 – Extrait de l'enregistrement du corpus "sweet home", où un être humain est filmé et enregistré en train d'effectuer une succession d'activités

Au terme de cette labellisation, nous avons pu obtenir une base contenant des exemples de signaux les plus variés possibles répondant à la taxonomie proposée (tableau 2.10).

Parole	Musique	Impact	Sons d'environnement	**Total**
27	30	16	17	**90**

TABLE 2.10 – Nombre d'échantillons par classes dans la base de donnée proposée.

2.4.2 Sélection et performance des caractéristiques

Le belief-KNN utilisant la distance euclidienne pour attribuer les croyances, il est nécessaire de prendre en compte la dynamique des données pour ne pas favoriser sans raison certaines caractéristiques. C'est pourquoi on procède à une normalisation classique caractéristique par

caractéristique, qui préserve les distances relatives au sein des classes considérées.
Un grand nombre de caractéristiques n'étant pas forcément une garantie d'une meilleure classification, on souhaite ici sélectionner les caractéristiques les plus pertinentes pour la classification, c'est à dire les caractéristiques qui séparent le mieux les classes individuellement.Pour ce faire on se propose d'utiliser une mesure simple de séparation de données. Pour chaque paire de classe et chaque caractéristique de la représentation finale du signal on calcule la quantité suivante :

$$D_{C_i,C_j} = \frac{D_{inter}(C_i, C_j)}{D_{intra}(C_i) + D_{intra}(C_j)} \tag{2.27}$$

où D_{intra} représente la distance moyenne intra-classe, et D_{inter} la distance moyenne inter-classe pour la caractéristique concernée.

$$D_{inter} = \frac{1}{N_i \cdot N_j} \sum_{k=1}^{N_i} \sum_{l=j}^{N_2} |x_k - y_l| \tag{2.28}$$

$$D_{intra} = \frac{2}{N_i \cdot (N_i - 1)} \sum_{k=1}^{N_i} \sum_{l=k}^{N_1} |x_k - y_l| \tag{2.29}$$

où N_i et N_j sont respectivement le nombre d'éléments x_k et y_l des classe C_i et C_j. Ainsi, on souhaite choisir les caractéristiques obtenant les plus haut D_{C_i,C_j}. Les caractéristiques donnant le meilleur score sont montrées dans le tableau 2.11. On sélectionne ainsi ces caractéristiques pour la classification.

	Parole	Musique	Impact	Son d'environnement
Parole	~	var(MFCC2)	LE	var(MFCC4)
Musique	var(MFCC2)	~	mean(MFCC5)	var(SF)
Impact	LE	mean(MFCC5)	~	mean(SR)
Son d'environnement	var(MFCC4)	var(SF)	mean(SR)	~

Table 2.11 – Meilleure caractéristique pour séparer les classes 2 à 2

On obtient donc comme représentation finale pour chaque signal un vecteur de dimension 6 : la variance des MFCC 2 et 4 (var(MFCC2), var(MFCC4)), la moyenne du MFCC 5 (mean(MFCC5)), le pourcentage de trames à basse énergie (LE), la variance du flux spectral (var(SF)) et la moyenne du rolloff spectral (mean(SR)). On a donc une caractéristique temporelle, deux caractéristiques "spectrales" et trois caractéristiques "transformées".

2.4.3 Résultats de classification

Pour la classification, on suit un processus de validation croisée. On divise la base de données en deux parties, une base d'apprentissage qui contiendra des éléments représentatifs qu'on utilisera pour rechercher les plus proches voisins, et une base de test qui permettra d'obtenir les performances de classification. La manière dont on répartit les éléments de la base de données est décrite dans le tableau 2.12.

Classe	**Nb d'éléments d'entraînements**	**Nb d'éléments de test**
Parole	11	16
Musique	11	19
Impact	5	11
Sons d'environnement	5	12
Total	**32**	**58**

TABLE 2.12 – Séparation des sons en une base d'entraînement et une base de test

On tire 1000 fois chacune des deux bases de manière aléatoire et procède à la classification de la base de test. Pour la prise de décision sur la classe de l'élément, [Den95] propose l'utilisation de probabilité pignistique, soit la transformation de la masse de croyance obtenue en probabilité en répartissant de manière équitable la masse donnée sur l'ignorance entre les différentes hypothèses qui composent l'espace de discernement. Dans notre cas, on choisit simplement l'hypothèse recevant le maximum de croyance, Ω inclus, nous donnant ainsi naturellement une "classe de rejet", ou classe de doute.

La base de sons dont nous disposons étant de petite taille, on s'intéresse au nombre d'éléments ayant été mal classifiés. Les résultats de la classification sont visibles dans le tableau 2.13, en comparaison avec un algorithme de classification classique de la littérature : la machine à vecteurs de support (Support Vector Machine, SVM). Pour les expériences suivantes nous avons choisi un nombre de voisins de 5, arbitrairement bas, pour rester en cohérence avec la taille de la base de données.

	Parole	**Musique**	**Impact**	**Environnement**	**Doute**	**Total**
Belief-KNN	1.4 (8.4%)	1.3 (6.8%)	0.7 (6.3%)	1.1 (9.1%)	4.1 (7%)	8.6 (14.8%)
SVM	0.6 (5.4%)	2.5 (13.1%)	2 (18.1%)	1.1 (9.1%)		6.3 (10.8%)

TABLE 2.13 – Résultats (nombre moyen de signaux mal classifiés)

Comme on peut le voir, les résultats des deux algorithmes sont comparables. Concernant le résultat global de classification, la machine à vecteurs de support produit de meilleurs résultats (10.8% de mauvaises classification contre 14.8% pour le belief-KNN). Cependant, il est important de prendre en compte la classe de doute apportée par le belief-KNN : ces sons classifiés comme "objet de doute" n'ont pas à être considérés comme de mauvaises classifications,

car ils atteignent précisément le but recherché (permettre au robot de différencier les cas où il est incapable de prédire la classe d'un élément). Ainsi, si on considère que la catégorie "doute" ne fait pas partie des mauvaises classifications, les performances du belief-KNN deviennent meilleures (4.5 échantillons mal classés en moyenne, soit 7,7% contre 10.8% pour la SVM). Ceci s'explique intuitivement de la manière suivante : lorsque les éléments à classifier font partie de classes très séparées linéairement, les algorithmes se comportent de manière similaire. Cependant, lorsqu'un élément présente des problèmes pour la classification (classes trop dispersées, voisins éloignés, plusieurs voisins de classes différentes), l'algorithme pourra produire des erreurs de classification, tandis que le belief-KNN aura tendance à classifier de tels éléments comme étant douteux.

L'objectif fixé en début de chapitre est donc atteint : la classification est performante, et les éléments impossibles à classifier sont ajoutés à la classe de doute, ce qui correspond au comportement attendu. En effet, le système étant destiné à être implémenté sur un robot, qui sera en mesure d'aller enquêter sur la nature du son entendu, une information partielle est préférable à une information erronée.

2.5 Conclusion

Dans ce chapitre un système de classification de sons en environnement intérieur a été proposé. Il se destine à être implémenté sur un robot compagnon, qui sera amené à naviguer dans un environnement dynamique contenant des humains. Pour ce faire, nous avons défini une taxonomie de classification de petite taille, qui pourrait permettre au robot de définir la priorité du son entendu. L'utilisation d'une méthode de classification utilisant les fonctions de croyance permet de classifier les éléments douteux en tant que tel, ce qui est souhaitable dans un système permettant d'affiner la perception à l'aide d'autres capteurs.

Il existe bien sûr des moyens d'améliorer le système proposé : une des difficultés rencontrées a été la création d'une base de données pour valider l'algorithme. La taxonomie proposée étant particulière, basée non sur la sémantique mais sur le type général de son entendu, il n'existe pas de base de données dans la littérature utilisable directement. Le système pourrait bénéficier d'un travail sur une base étiquetée de très grande taille, permettant d'explorer plus précisément les défauts d'une telle classification. Il pourrait être aussi intéressant de travailler sur la classification en elle-même et plus précisément sur la gestion du conflit : [LPD13] s'intéresse par exemple à la création d'une nouvelle règle de combinaison pour le belief-KNN, en utilisant la distance du voisin considéré à l'élément à classifier pour créer deux types de masses : une attribuant l'élément à classifier à la classe de ce voisin, et une autre exprimant la croyance en le fait que l'élément à classifier n'appartient pas à la classe de ce voisin. De manière plus simple, on peut imaginer de rejeter toute forme de conflit sur l'ignorance. Toutefois, cela ne permettrait de classifier de manière juste que les éléments ayant un nombre de voisins appartenant très majoritairement à une classe.

Ce système est adapté à un robot compagnon : en effet ces robots sont souvent équipés de microphones. Malheureusement, ces microphones sont généralement de mauvaises qualité

(ces capteurs étant encore coûteux actuellement). De plus, un robot compagnon étant mobile, il est équipé de moteurs, qui peuvent perturber facilement l'audition du robot et qui font de l'audio une modalité difficile à utiliser. Une taxonomie à la sémantique peu précise est plus intéressante, car elle permet d'utiliser la modalité comme un guide pour le robot plutôt que comme une source d'informations précise.

Chapitre 3

Fusion audiovisuelle pour la détection de locuteurs successifs dans une conversation

Sommaire

3.1 Introduction 41
3.2 État de l'art sur la fusion multimodale 43
3.2.1 Types de fusion 43
3.2.2 Méthodes de fusion 45
3.3 Détection de locuteurs successifs par fusion audiovisuelle probabiliste 51
3.3.1 Présentation de l'architecture 51
3.3.2 Détection de visages dans la vidéo par fusion probabiliste instantanée . . 53
3.3.3 Localisation de la source sonore 60
3.3.4 Fusion audiovisuelle à l'aide d'un filtre temporel probabiliste 63
3.4 Résultats 69
3.4.1 Présentation du robot Reeti 69
3.4.2 Résultats expérimentaux 69
3.5 Conclusion 76

3.1 Introduction

Ce chapitre décrit un travail qui s'inscrit dans le cadre de la perception robotique. Dans le chapitre 2, l'importance de la modalité audio pour un robot compagnon a été abordée. Comme on l'a dit, cette modalité apporte deux types d'information : le contenu sémantique ("Qu'ai je entendu ?"), qui a été exploré au chapitre 2, et la position de la source sonore ("D'où venait ce son ?"). C'est cette deuxième information qu'on cherche ici à exploiter, pour détecter les locuteurs successifs au sein d'une conversation. Pour cela, on propose de fusionner les modalités audio et vidéo.

Comme décrit chapitre 1, un robot compagnon sera amené à interagir avec les humains de son environnement. Cela implique pour le robot de se positionner de manière à interagir de façon "naturelle" : il a été montré que les humains ont tendance à se positionner naturellement

en cercle dont la taille dépend du nombre d'acteurs [RMSL15] en cas d'interaction. Ceci permet aux différents acteurs d'un groupe d'avoir un accès visuel facile aux autres membres du groupe.

Dans la majorité des cas, un robot inclus comme membre d'un groupe interagissant n'aura pas le même impact sur l'interaction qu'un membre humain : en effet, interagir naturellement au sein d'une conversation est encore un problème ouvert, et un robot équipé de capacités de conversation égales à celles d'un humain n'a pas encore été mis au point. Cependant, il reste intéressant pour un robot d'avoir un comportement d'écoute similaire à celui d'un être humain : lorsqu'une personne assiste à une interaction ou un échange, certains éléments de la scène vont plus souvent attirer son regard. [MGP09] montre que les visages sont des attracteurs de l'attention pour un être humain. Lorsque les humains présents sont silencieux, les visages attirant le plus l'attention sont ceux qui bougent le plus, le mouvement étant un autre attracteur de l'attention. Cependant, lorsqu'une personne de la scène est en train de parler, c'est son visage qui attire directement l'attention du spectateur, comme le montre [SPG13]. De la même manière qu'il est naturel pour un être humain d'être attiré par le visage du locuteur courant dans une conversation, il semble logique qu'un robot porte son attention sur la personne en train de parler lorsqu'il assiste à une scène sociale. On propose donc ici de réaliser un système de détection de locuteurs successifs dans une conversation, à l'aide d'un robot équipé d'une caméra RGB classique et d'une paire de microphones.

On pose donc le problème ainsi : un robot assiste à une interaction entre 2 personnes ou plus (figure 3.1). A chaque instant, on souhaite que le robot soit capable de détecter le visage du locuteur actuel de l'interaction, si celui-ci se trouve dans son champ de vision. A tout moment un nouveau locuteur, visible ou non, peut se joindre à la conversation, tandis qu'un autre peut la quitter (et donc quitter le champ visuel du robot). La scène peut être soumise à une illumination variable.

FIGURE 3.1 – Exemple de situation où le robot est témoin d'une interaction

Pour cela, nous proposons un système de fusion audiovisuelle, reposant sur l'extraction de features simples sur chacune des deux modalités. Sur la vidéo, les visages sont extraits à l'aide d'un filtre bayésien naïf apprenant la couleur de la peau en ligne. Sur le son, la

source est localisée à l'aide d'une méthode reposant sur la corrélation croisée entre les signaux issus des deux microphones. Ces deux informations sont ensuite fusionnées au sein d'un filtre bayésien temporel, calculant la distribution de probabilités a posteriori sur l'espace de positions possibles du locuteur actuel. Ces travaux ont fait l'objet d'une publication dans la conférence VISAPP 2014[Lab+14].

Ce chapitre est divisé de la manière suivante : un état de l'art sur les méthodes de fusion multimodale est proposé en section 3.2, puis l'architecture proposée est brièvement introduite en section 3.3.1. Les traitements capteurs sont respectivement décrits en section 3.3.2 pour la vidéo et 3.3.3 pour l'audio. Finalement la section 3.3.4 décrit la méthode de fusion employée, et la section 3.4 montre des résultats de l'algorithme dans son intégralité.

3.2 État de l'art sur la fusion multimodale

La fusion de données multimodales est un sujet vaste qui a été l'objet de travaux dans une grande variété de domaines. Le problème est le suivant : on souhaite estimer l'état réel x_t d'un système à chaque instant t (dans notre cas, la position du locuteur dans une conversation par rapport à un robot). Pour ce faire, on dispose de K sources de natures différentes observant l'état de ce système (dans notre cas des capteurs vidéos et audio). On a donc à notre disposition :

— Un espace d'états $\mathcal{X} = \{X^1, \cdots, X^N\}$ dans lesquels peut se trouver le système
— Des observations sur l'état du système à chaque instant d'observation. Un capteur i, pour i ∈ [1,K] fournit donc la succession d'observations $z^i = \{z_1^i, \cdots, z_t^i\}$

L'objectif est ici de combiner l'information apportée par ces différentes sources pour décider de l'état réel du système. Les sources peuvent produire de l'information incertaine (erreurs de mesure) et asynchrone (temps d'acquisition différents).

Cette section présente donc un bref état de l'art sur la fusion de données multimodales. Le sujet ayant été exploré sous de multiples facettes dans la littérature, comme le résume [Atr+10], on divise cette section en deux aspects principaux de la fusion :

— Le type de fusion : c'est le niveau auquel la fusion a lieu entre les différentes modalités. Ceci correspond donc à l'architecture de fusion proposée.
— La méthode de fusion : c'est la manière dont les informations des différentes modalités sont combinées pour obtenir l'état du système.

3.2.1 Types de fusion

Il existe deux grands types d'architecture de fusion de données à distinguer : la fusion précoce, ou "fusion niveau caractéristiques" et la fusion tardive, ou "fusion niveau décision" [HL97 ; SWS05]. On décrit ici très brièvement ces deux genres de fusion.

3.2.1.1 Fusion précoce

La fusion précoce consiste à utiliser directement les caractéristiques extraites des modalités à fusionner pour prendre une décision sur l'état du système. Il n'y a dans ce cas pas de niveau intermédiaire de décision modalité par modalité. Par exemple, dans notre cas, cela signifierait qu'aucune décision n'est prise sur la position possible du locuteur dans la conversation. Des caractéristiques sont justes extraites de la vidéo (telles que la présence de mouvement) et de l'audio (présence de son), et sont directement fusionnées. Ce procédé est illustré par la figure 3.2.

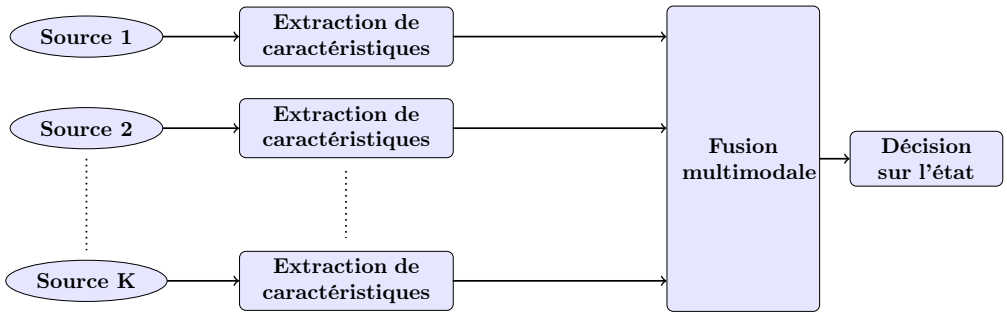


FIGURE 3.2 – Illustration du processus général de fusion précoce : la fusion est appliquée directement sur des caractéristiques extraites de différentes modalités

3.2.1.2 Fusion tardive

La fusion tardive consiste à fusionner non pas les caractéristiques extraites directement des sources, mais des décisions prises sur ces sources. Dans ce cas, on apprend des concepts sémantiques directement au niveau unimodal. Par exemple, dans notre cas, cela revient à détecter directement des locuteurs possibles au niveau de la vidéo et de l'audio (l'information de ce qu'est un locuteur est donc connue au niveau de ces modalités), et de fusionner ces deux informations. Le processus de fusion tardive est illustré par la figure 3.3

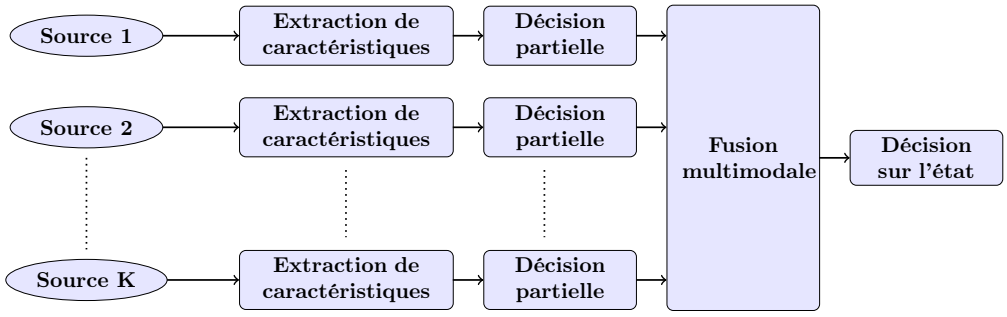


FIGURE 3.3 – Illustration du processus général de fusion tardive : la fusion est appliquée sur un ensemble de décision prise au niveau unimodal

3.2.2 Méthodes de fusion

De nombreuses méthodes provenant de différents domaines ont été utilisées pour répondre au problème de fusion multimodale. Les méthodes les plus populaires restent cependant les méthodes probabilistes : elles sont la plupart du temps simples à mettre en place, performantes, et peu coûteuses en ressources de calcul. Elles sont aussi particulièrement adaptées aux problèmes de fusion temporelle. C'est pour cette raison que nous avons utilisé une fusion probabiliste pour répondre au problème que l'on se pose. On propose ainsi de détailler ce type de méthodes dans un premier temps. Dans un second temps, on décrira brièvement les autres types de méthodes de fusion existant dans la littérature.

3.2.2.1 Méthodes probabilistes

Cette catégorie repose sur la théorie d'estimation d'état probabiliste. Dans cette situation l'objectif est d'estimer une distribution de probabilités sur l'espace d'états. Deux types de fusion probabiliste existent : la fusion instantanée, qui prend uniquement en compte les observations à un instant t donné pour fusionner l'information, et la fusion temporelle, qui prend en compte l'information aux instants précédents pour estimer l'état du système à l'instant t.

Fusion instantanée

Cette méthode permet de fusionner les observations à un instant donné, sans prendre en compte leur évolution temporelle. Elle repose sur une application simple de la règle de Bayes, qui s'exprime pour deux variables aléatoires X et Y de la manière suivante :

$$P(X|Y) = P(X)\frac{P(Y|X)}{P(Y)} \tag{3.1}$$

Cette règle nous permet de calculer la probabilité d'un évènement conditionné à un autre. Il est possible d'étendre le conditionnement à plusieurs variables aléatoires. Par exemple, pour 3 variables aléatoires X,Y,Z quelconques :

$$P(X|Y,Z) = P(X|Z)\frac{P(Y|X,Z)}{P(Y|Z)} \tag{3.2}$$

Dans un cas d'indépendance conditionnelle entre les variables Y et Z, il est possible d'écrire :

$$P(Y|X,Z) = P(Y|X) \tag{3.3}$$
$$P(Y|Z) = P(Y) \tag{3.4}$$

Ainsi, on peut écrire le conditionnement de X par Y et Z de la manière suivante, sous condition d'indépendance conditionnelle, et en utilisant 2 fois la règle de Bayes :

$$P(X|Y,Z) = P(X)\frac{P(Y|X)P(Z|X)}{P(Y)P(Z)} \tag{3.5}$$

L'application de cette règle à K sources d'observations dans le cadre d'une fusion multimodale nous permet d'écrire directement :

$$P(x_t|z_t^1,\cdots,z_t^K) \propto P(x_t)\prod_{j=1}^{K}\Big(P(z_t^j|x_t)\Big) \tag{3.6}$$

Cette proportionnalité vient du fait que le terme se trouvant au dénominateur de la fraction, $\prod_{j=1}^{K}\Big(P(z_t^j)\Big)$, sera fixe pour tout état $x_t \in \mathcal{X}$. Ainsi, la règle de Bayes permet d'estimer l'état du système, sous deux conditions simples, mais fondamentales :

— **L'hypothèse d'indépendance conditionnelles des observations :** cette hypothèse est généralement acceptée dans le cadre de la fusion. Elle simplifie le calcul de l'état, et se trouve souvent vérifiée, particulièrement dans le cas multimodal, où les observations sont de natures et de sources complètement différentes, ce qui garantit leur indépendance.
— La connaissance des distributions $\mathrm{P}(z_t^j|x_t)$, communément appelés **modèles capteurs**. Ces modèles capteurs, correspondent à la probabilité d'obtenir l'observation z_t^j, si le système se trouve dans l'état $x_t \in \mathcal{X}$. Ces modèles peuvent être appris à partir de données réelles, ou fixés par des experts connaissant les capteurs.

Pour décider de l'état du système, la méthode généralement adoptée est d'utiliser le maximum a posteriori, soit :

$$x_t^{\text{final}} = \underset{x_t\in\mathcal{X}}{\text{argmax}} P(x_t|z_t^1,\cdots,z_t^K) \tag{3.7}$$

Ce genre de méthodes a été employé pour des applications variées, telles que la reconnaissance de paroles en audiovisuel [Pap+09], la surveillance multimodale [AKJ06], ou encore la détection d'évènements dans des vidéos [XC06].

Fusion temporelle

La fusion temporelle probabiliste correspond à l'estimation de l'état x_t du système, prenant en compte les observations à l'instant t, mais aussi l'estimée de l'état x_{t-1} à l'instant précédent, soit dans le cas d'une unique source d'observation la distribution de probabilités $\mathrm{p}(x_t|z_t, x_{t-1})$. Il repose sur l'estimation d'un filtre bayésien, qui comporte deux étapes séparées : la prédiction de l'état x_t, connaissant l'état x_{t-1} à l'instant précédent, puis la correction

à l'aide des observations à l'instant t. On illustre dans cette partie le cas d'une fusion temporelle pour une unique source d'observation z. Elle peut être étendue facilement pour le cas multimodale. Dans la suite de ce chapitre, on utilisera de manière interchangeable les termes "fusion probabiliste temporelle", "filtre probabiliste temporel", et "filtre bayésien".

L'étape de prédiction s'écrit comme suit :

$$\underbrace{p(x_t|z_{t-1})}_{\text{Prédiction}} = \int_{x_{t-1}} \left(\underbrace{p(x_t|x_{t-1})}_{\text{Modèle dynamique}} \cdot \underbrace{p(x_{t-1}|z_{t-1})}_{\text{Estimée à t-1}} \right) \tag{3.8}$$

Cette étape nécessite la connaissance de la distribution $p(x_t|x_{t-1})$. Cette distribution est généralement appelée **modèle dynamique**. Il est possible d'utiliser une telle formule de prédiction car on fait généralement une hypothèse de Markov, qui stipule que l'état x_t à l'instant t ne dépend que de l'état à l'instant précédent d'observation. Les états aux instants antérieurs ne sont pas considérés. La manière d'instancier le modèle dynamique dépend de l'application à laquelle le système se destine. Dans certains cas, ce modèle peut être appris, ou fixé par des experts, de la même manière que le modèle capteur montré dans le cas de la fusion instantanée.

L'étape de correction par les observations s'écrit comme suit :

$$\underbrace{p(x_t|z_t)}_{\text{Estimée à l'instant t}} \propto \underbrace{p(z_t|x_t)}_{\substack{\text{Correction par} \\ \text{modèle capteur}}} \cdot \underbrace{p(x_t|z_{t-1})}_{\text{Prédiction}} \tag{3.9}$$

Tout comme dans le cas de la fusion instantanée, elle nécessite la connaissance de la distribution $p(z_t|x_t)$, qui correspond au **modèle capteur**. Ici encore, il peut être appris sur des données réelles ou fixé par un expert. Le processus du filtrage temporel probabiliste est illustré par la figure 3.4.

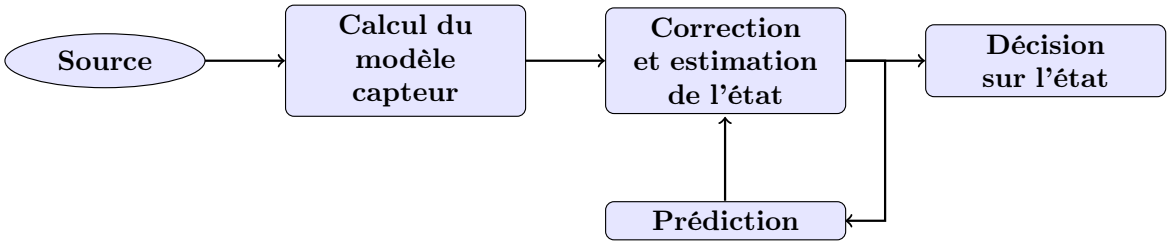


FIGURE 3.4 – Illustration de la fusion probabiliste temporelle

Ce filtre temporel, particulièrement adapté au traitement de flux capteur, a été implémenté sous différentes formes :

— Le filtre de Kalman : dans cette situation, on considère que le bruit lié aux observations,

ainsi qu'au modèle dynamique est gaussien, et le filtre bayésien dans ce cas à estimer à chaque instant la moyenne et la variance d'une gaussienne. Ce filtre a été utilisé dans de nombreuses applications de suivi de cible et de fusion, comme c'est le cas par exemple de [LGG04] qui utilise ce filtre pour estimer un mouvement à partir de caractéristiques audio et vidéo.

— Le filtre à particule [Aru+02] : ce filtre permet d'estimer des distributions de probabilités non paramétriques, et est devenu au cours des dernières années particulièrement populaire dans les problèmes d'estimation. Dans ce cas, les particules sont des états possibles du système, caractérisés par un poids représentant leur probabilité. De la même manière que le filtre bayésien classique, deux étapes ont lieu : tout d'abord, des particules sont tirées au hasard à partir du modèle dynamique et des particules à l'instant précédent, puis leur poids est mis à jour à l'aide du modèle d'observation.

Ces filtres temporels sont utilisés très couramment dans la fusion de données : ils permettent d'estimer l'état du système, et prennent en compte l'état aux instants précédents pour faire du suivi, ce qui est précieux dans le cadre de l'analyse de flux capteur.

3.2.2.2 Autres méthodes de fusion

Cette section présente brièvement les différents types de méthodes exploitées pour la fusion de données multimodales. Pour plus de précisions sur chacune des catégories présentées ici, on pourra par exemple se référer à [Atr+10] et [Kha+13].

Méthodes basées sur des règles

Cette catégorie contient les méthodes basées sur des règles statistiques de combinaison. Elles incluent les règles les plus simples telle que les règles de logique (maximum, minimum, ET logique, OU logique) entre les différentes modalités. On trouve aussi des combinaisons telles que la fusion pondérée linéaire. Elle consiste en une somme pondérée, ou un produit des observations. Elle peut se faire aux deux niveaux de fusions décrit précédemment, précoce ou tardifs. Dans la suite on considèrera que les différents z^i représente les informations à fusionner, indépendamment de leur type. Ainsi une fusion pondérée linéaire peut se traduire par une somme ou un produit :

$$x_{final} = \sum_{i}^{K} \left(w_i \cdot z^i \right) \tag{3.10}$$

$$x_{final} = \prod_{i}^{K} \left(z^i \right)^{w_i} \tag{3.11}$$

où les w_i représentent les poids attribués aux différentes observations.

[INN03] propose par exemple une méthode de détection de monologues dans du contenu audiovisuel de type film reposant sur une simple somme pondérée de 3 facteurs : présence d'un

visage à l'écran, présence de parole, et synchronie entre l'audio et la vidéo. Cette somme est seuillée et une décision est prise pour définir si un monologue a lieu à l'écran.

Une autre règle de combinaison utilisée est le vote majoritaire. Dans ce cas là, c'est l'état obtenant le plus de vote de la part de classifieurs de modalité qui remporte le vote [RP97].

Ces méthodes ont l'avantage d'être simples à mettre en place. Cependant elles ne sont pas adaptées à une fusion temporelle, et ne prennent pas en compte une éventuelle incertitude des données. Les observations sont directement utilisées pour prendre la décision. Elles sont le plus souvent utilisées pour de la fusion instantanée.

Méthode de classification

Comme leur nom l'indique, ces méthodes sont celles qui considèrent la fusion multimodale comme un problème de classification. Dans ce cas, la fusion consiste en l'application d'un algorithme de classification sur l'ensemble des informations apportées par les différentes sources, afin de déterminer la classe, correspondant à l'état, du système.

Un des algorithmes les plus utilisés pour la fusion multimodale est la machine à vecteurs de support (SVM) [SS01], dont les applications sont nombreuses : [BC07] propose par exemple une méthode d'identification audiovisuelle de personne destinée à des systèmes de sécurité. Ici une SVM est entrainée sur un score d'association entre évènements audio et vidéo. [Ada+03] utilise quant à lui les résultats de classifieurs intermédiaires appliqués à de la vidéo, de l'audio et du texte, comme données d'entraînement pour une SVM, dans le but de reconnaître des concepts sémantiques sur des vidéos (tels que du feu, un paysage extérieur, etc...).

Ces méthodes sont principalement des méthodes de fusion instantanée. Elles présentent des avantages, car elles permettent l'utilisation d'algorithmes tirés de la littérature de l'apprentissage automatique. Cependant elles sont souvent coûteuses en ressources de calcul, ce qui rend difficile leur utilisation en cadence capteur.

Méthodes floues

La logique floue a été introduite par [Zad65] en 1965 et consiste à définir un cadre formel de raisonnement avec des concepts flous. La logique en général s'appuie sur la théorie des ensembles pour estimer l'état d'un système. Ainsi, dans la logique classique, l'espace d'états est considéré comme un ensemble. Pour estimer l'état, on peut attribuer à chaque sous-ensemble de $\mathcal{X}$ une valeur de vérité binaire : vrai ou faux. Dans ce cas, pour fusionner des informations binaires, on retrouve les méthodes basées sur des règles (on peut procéder à un vote de majorité : le sous-ensemble ayant reçu le plus de vote vrai remporte le vote et est considéré comme l'état estimé).

La logique floue introduit la notion de sous-ensemble flou. Il est défini comme suit : soit X un ensemble. Un sous-ensemble flou A de X est caractérisé par une fonction d'appartenance, généralement notée μ_A définie sur X à valeur dans l'intervalle [0,1] :

$$\begin{array}{rccc} \mu_a & : & X & \mapsto & [0,1] \\ & & x & \mapsto & \mu_A(x) \end{array} \tag{3.12}$$

Cette notion de sous-ensemble flou permet de s'affranchir d'une valeur de vérité binaire, et de définir un degré d'appartenance à un sous-ensemble (un état). Ainsi on peut attribuer à chaque source d'observation z_i une fonction d'appartenance μ_{z_i}. Pour combiner l'information apportée par les différentes sources, les opérateurs de la logique classique ont été adaptés.

Ces méthodes sont utilisées tant pour la fusion instantanée, via la combinaison de fonctions d'appartenance que pour la fusion temporelle.

Méthodes crédibilistes

Comme on l'a vu dans la section 2.3.2, le cadre évidentiel est une extension du domaine probabiliste pour augmenter le niveau de représentation disponible. Dans ce cas là, on représente la connaissance sur l'ensemble des sous-ensembles de l'espace d'états $2^{\mathcal{X}}$. Cette manière de procéder permet de modéliser explicitement certaines formes d'incertitude liée aux observations. Ainsi, de manière similaire à la logique floue, dans ce cas, à chaque observation on peut associer une fonction de croyance $\mathrm{m}^{\mathcal{X}}_{z_i}$ correspondant à la croyance apportée par la source i.

Une fois ces distribution de masses (BBA) créées, il existe différentes manières de les combiner, la plus connue étant la combinaison conjonctive (section 2.3.2) :

$$m^{\mathcal{X}}_{z_i \textcircled{\cap} z_j}(H) = (m^{\mathcal{X}}_{z_i} \textcircled{\cap} m^{\mathcal{X}}_{z_j})(H) = \sum_{A \cap B = H} \Big(m^{\mathcal{X}}_{z_i}(A) \cdot m^{\mathcal{X}}_{z_j}(B) \Big) \tag{3.13}$$

Comme on l'a expliqué, cette représentation a l'avantage de modéliser l'ignorance, et de pouvoir attribuer de la croyance à n'importe quel sous-ensemble de $\mathcal{X}$. Dans le cas où il serait nécessaire de prendre une décision sur un état unique, il existe des méthodes de prise de décision, la plus courante étant l'utilisation des probabilités pignistiques [Sme05]. Dans ce cas la fonction de croyance $\mathrm{m}^{\mathcal{X}}$ est transformée en probabilité pignistique BetP, de la manière suivante, pour tout $\mathcal{X}^i \in \mathcal{X}$.

$$\mathrm{BetP}\{m^{\mathcal{X}}\}(\mathcal{X}^i) = \frac{1}{(1 - m^{\mathcal{X}}(\emptyset))} \sum_{H \subseteq \mathcal{X}, \mathcal{X}^i \in \mathcal{X}} \frac{m^{\mathcal{X}}(H)}{|H|} \tag{3.14}$$

La décision finale est généralement prise en choisissant l'élément $\mathcal{X}^i$ possédant la plus grande probabilité pignistique.

$$x^{\text{final}} = \underset{\mathcal{X}^i \in \mathcal{X}}{\operatorname{argmax}} \ \mathrm{BetP}\{m^{\mathcal{X}}\}(\mathcal{X}^i) \tag{3.15}$$

La théorie crédibiliste a été utilisée dans différents cadres d'application en fusion multimodale : [Hon+09] propose par exemple l'utilisation du cadre crédibiliste pour la fusion instantanée multi-capteurs dans des maisons intelligentes. En établissant une correspondance évidentielle entre des données capteurs et des concepts sémantiques, ce travail permet de reconnaître des activités humaines à partir de n'importe quel type de capteur transmettant de l'information pertinente. [RB10] propose une fusion crédibiliste de signaux électroencéphalographiques et de la reconnaissance de gestes humains pour l'interaction homme-robot. De manière générale, la fusion crédibiliste temporelle commence à se populariser dans le monde de la robotique, notamment pour la navigation [Kur+15 ; Lab+16], comme on le verra en chapitre 5.

Cette catégorie de méthode présente les avantages exposés au chapitre précédent : elle permet une modélisation de la connaissance disponible plus proche de celle d'un être humain, et propose des techniques intuitives de manipulation de la croyance. Cette augmentation du pouvoir de représentation se fait toutefois au prix d'une augmentation du coût de calcul.

3.3 Détection de locuteurs successifs par fusion audiovisuelle probabiliste

3.3.1 Présentation de l'architecture

On considère un robot équipé d'une caméra RGB classique et d'une paire de microphones. Ce robot souhaite détecter le locuteur dans une conversation à chaque instant d'observation. On peut résumer le robot "physiquement" comme montré par la figure 3.5 : une vue de dessus du "robot", représenté par une caméra placée à égale distance de deux microphones.

Les personnes présentes dans la scène sociale à laquelle le robot assiste ne sont pas forcément dans le champ de vision de celui-ci, et peuvent être silencieuses ou en train de parler.

L'architecture que nous proposons est visible dans la figure 3.6. Elle consiste en une fusion de données homogènes extraites des flux capteurs du robot, et s'approche par conséquent d'une fusion de type précoce. Elle se divise en trois grandes parties :

- **Détection de visages sur le flux vidéo :** Pour détecter les visages, nous avons choisi de détecter les pixels de chaque image ayant la couleur de la peau. Pour réaliser cette détection, nous avons utilisé une fusion probabiliste instantanée sur les 3 canaux couleurs de la vidéo.
- **Localisation de sons sur le flux audio :** La source sonore est localisée à chaque instant d'observation, pour permettre au robot de déterminer la position angulaire probable du locuteur.
- **Fusion probabiliste temporelle pour la détection du locuteur :** Les informations apportées par l'audio et la vidéo sont fusionnées au sein d'un filtre probabiliste temporel, permettant ainsi le suivi de locuteur dans le temps.

Les traitements audio et vidéo pouvant dépendre beaucoup des conditions d'acquisition du

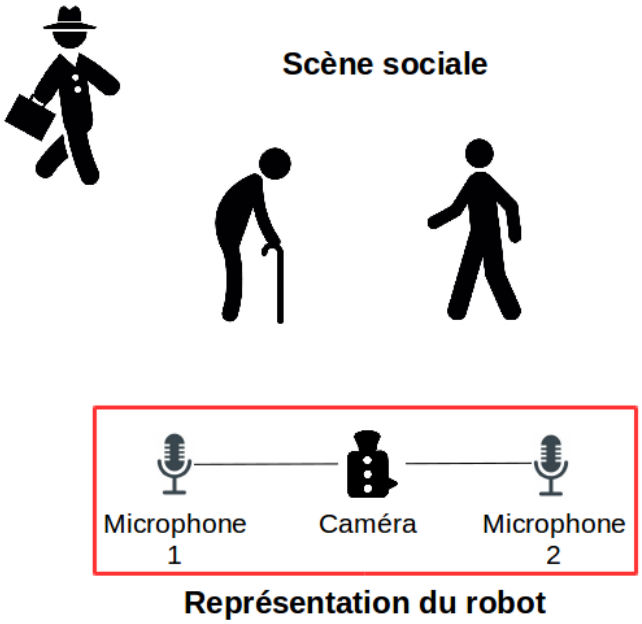


FIGURE 3.5 – Représentation du robot face à une scène sociale

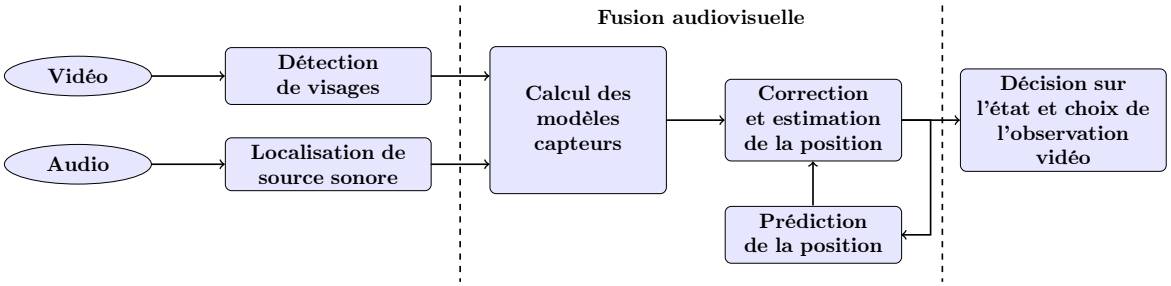


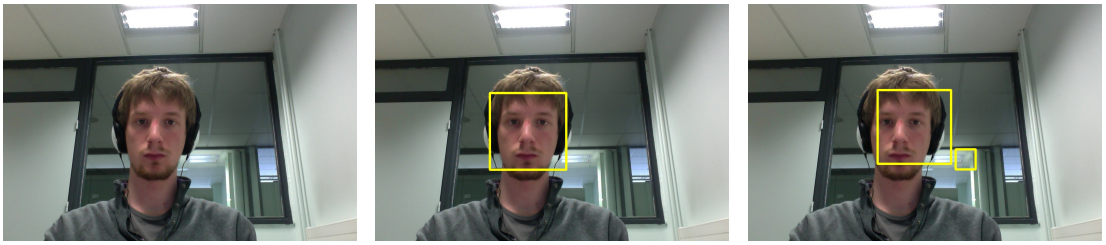
FIGURE 3.6 – Architecture de l'algorithme proposé ici

robot, nous faisons quelques hypothèses pour définir le cadre dans lequel s'inscrit ce travail :

- On considère ici que le robot est déjà positionné en position d'interaction. Il est donc témoin d'une interaction ayant lieu face à lui, et est positionné de manière à avoir, au moins au début de l'interaction, des humains y participant dans son champ de vision
- On considère qu'une seule personne parle à la fois dans la conversation. Cela permet de s'affranchir du problème de séparation de sources, qui est calculatoire, et difficile à résoudre avec uniquement deux microphones.
- L'environnement du robot est considéré comme silencieux, en dehors de cette interaction. Cette hypothèse s'inscrit dans la même idée que la précédente : une séparation de source est exclue, et extraire la localisation de l'information pertinente dans un contenu sonore trop riche est difficile. Cependant, l'égo-bruit du robot, le bruit inhérent aux microphones et la possibilité d'un "brouhaha" de fond sont pris en compte : on procède à un filtrage du signal en pré-traitement permettant d'éliminer en partie le bruit.

3.3.2 Détection de visages dans la vidéo par fusion probabiliste instantanée

Un des algorithmes les plus utilisés pour la détection de visages en vision par ordinateur est l'algorithme de Viola & Jones [VJ01]. Ses performances et sa mise à disposition via des bibliothèques de vision par ordinateur (OpenCV) en ont fait une des méthodes les plus populaires (l'article [VJ01] a été cité à ce jour plus de 12000 fois). Cependant, cet algorithme, bien que performant, présente des défauts : tout d'abord, il est très sensible à des paramètres tels que l'orientation du visage ou le point de vue, comme le montre la figure 3.7. Deux classifieurs nativement disponibles ont été entraînés : un classifieur de détection frontale (pour les visages de face) et un classifieur de détection de profil.



(a) Image frontale (b) Détection frontale (c) Détection de profil

(d) Profil (e) Détection frontale (f) Détection de profil

(g) Profil 2 (h) Détection frontale (i) Détection de profil

FIGURE 3.7 – Quelques exemples de détection par l'algorithme de [VJ01]. La première colonne contient les images originales, et la deuxième et troisième respectivement la détection à l'aide de l'apprentissage sur des visages de face et de profil

Sur ces exemples on distingue des problèmes : fausses détections (figures 3.7c et 3.7i) mais surtout des faux négatifs (figures 3.7h et 3.7i). Des tests sur des vidéos dans différentes conditions montrent que cet algorithme n'est pas idéal pour de la détection trame par trame

en continu lors de vidéo : les différences de conditions d'illumination, point de vue, etc font souvent "décrocher" l'algorithme d'un visage précédemment détecté. De plus, s'appuyer sur un détecteur reposant sur des caractéristiques morphologiques n'est pas adapté à des capteurs de faible qualité. Finalement, cet algorithme reste coûteux en terme de temps de calcul et n'est pas adapté à un flux d'image à grande cadence, surtout si on considère le fait que les ressources d'un robot amené à utiliser plusieurs fonctions en même temps sont précieuses.

On souhaite donc être en mesure de détecter des visages rapidement (cadence vidéo), avec le moins de faux négatifs (détections manquées) possible. Un bon indicateur de la présence d'un visage est la peau. On se propose donc de détecter les pixels de couleur de la peau dans l'image. Pour ce faire, on a à notre disposition 3 informations différentes à fusionner : l'information de couleur apportée par les 3 canaux couleurs de la vidéo, le rouge, le vert, et le bleu.

On utilise la fusion instantanée probabiliste, qui a pour avantage d'avoir un temps de calcul faible pour de bonnes performances dans de nombreux cas [VSA03], malgré son hypothèse d'indépendance conditionnelle des sources qui n'est pas toujours vérifiée dans le cas de données réelles.

La fusion instantanée proposée ici est divisée en plusieurs étapes, comme le montre la figure 3.8 : dans un premier temps, une base de pixels de peau est créée, permettant d'avoir des exemples réels de pixels de la couleur recherchée. Pour obtenir un algorithme une détection adaptée à la scène et aux personnes présentes lors de l'interaction, on propose de créer une base d'apprentissage en ligne à l'aide de l'algorithme de Viola & Jones. Cette base est ensuite utilisée pour apprendre les modèles capteurs, comme on l'a décrit en section 3.2.2.1. Finalement, une inférence est faite sur chaque pixel de l'image pour déterminer s'il appartient ou non à la catégorie "peau".

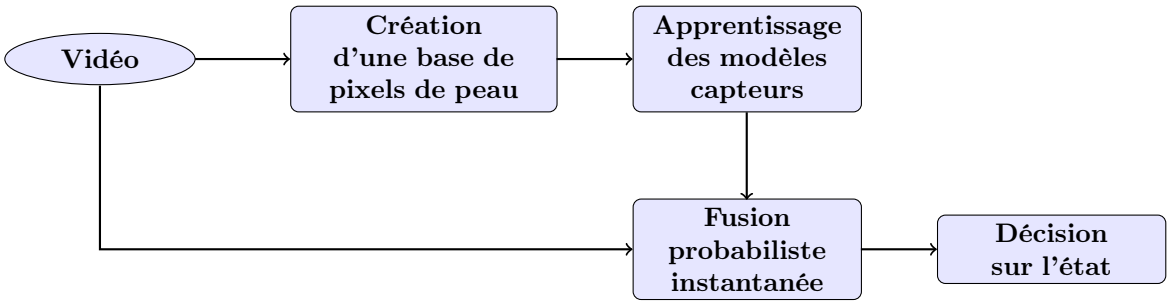


FIGURE 3.8 – Processus de détection de peau pour un pixel de l'image

3.3.2.1 Fusion probabiliste instantanée pour la détection des pixels de peau dans l'image

Comme on l'a dit précédemment, on cherche ici à attribuer à chaque pixel une étiquette : "peau" ou "non-peau". Pour attribuer cette étiquette, on dispose de l'information de couleur

sur trois canaux : le rouge (R), le vert (V) et le bleu (B), chacune de ces 3 valeurs étant codée sur 8 bits, donc comprise entre 0 et 255.

On cherche donc pour chaque pixel de l'image à estimer la distribution de probabilités P(C|R,V,B), où C $\in \{c, \bar{c}\}$, avec

— c : le pixel considéré est un pixel de peau

— $\bar{c}$: le pixel considéré n'est pas un pixel de peau

On fait ici une hypothèse d'indépendance conditionnelle entre les 3 canaux couleur de la vidéo. Ceci nous permet donc d'écrire, comme on l'a montré en équation 3.6 :

$$P(c|R,V,B) \propto P(c) \cdot P(R|c) \cdot P(V|c) \cdot P(B|c) \tag{3.16}$$

$$P(\bar{c}|R,V,B) \propto P(\bar{c}) \cdot P(R|\bar{c}) \cdot P(V|\bar{c}) \cdot P(B|\bar{c}) \tag{3.17}$$

L'hypothèse d'indépendance des trois canaux de couleur rouge, vert et bleu n'est pas vérifiée dans la réalité. Cependant une telle approximation n'entrave pas les résultats généraux de la méthode. Le score final d'appartenance à la catégorie des échantillons positifs pour un pixel est finalement donné par :

$$Score = \frac{P(c|R,V,B)}{P(\bar{c}|R,V,B) + P(c|R,V,B)} \tag{3.18}$$

Ce score, entre 0 et 1 correspondra à un échantillon positif s'il est proche de 1 et négatif s'il proche de 0.

Comme on l'a dit précédemment, pour obtenir ce score, il est nécessaire de connaître les distribution de probabilités P(R|C), P(V|C) et P(B|C). Ce sont les modèles capteurs de notre problème. Ils peuvent être fixés manuellement ou appris à partir des données. C'est cette deuxième solution que l'on souhaite mettre en œuvre, car elle permet au robot de s'adapter à la scène à laquelle il assiste.

Pour apprendre ces modèles à partir des données, on a donc besoin d'exemples de pixels étiquetés. On souhaite ainsi construire une base de pixels pour l'apprentissage, où pour chaque pixel 4 informations sont disponibles : R,V,B, et C. Pour permettre au système d'être adaptable, ces pixels doivent être générés directement depuis la scène à laquelle assiste le robot, et doivent pouvoir être remis à jour à volonté. On propose d'utiliser pour cela le détecteur de visage de [VJ01].

3.3.2.2 Création de la base de pixels de peau

Comme on l'a dit précédemment, la couleur de la peau sur une vidéo en couleur est sujette à des changements qui dépendent des conditions d'acquisition : changement d'illumination, de position des personnes, ombres, etc... Il est donc judicieux d'adapter l'apprentissage des modèles capteurs à la scène à laquelle le robot assiste. Pour ce faire, on propose la création

d'une base d'apprentissage en ligne, directement sur le flux vidéo acquis par le robot. On utilise pour cela l'algorithme de Viola & Jones.

L'algorithme proposé par [VJ01] est ainsi appliqué sur les premières trames provenant du flux vidéo. Le but ici est d'obtenir des pixels candidats pour des visages, pour apprendre leur couleur. L'algorithme de [VJ01] étant performant ponctuellement, il est possible de détecter un visage au cours des premières trames, et d'obtenir des pixels pour l'apprentissage.

Pour limiter le nombre de fausses détections, on contrôle la taille des détections proposées par l'algorithme de Viola & Jones. En effet, les fausses détections données par cet algorithme sont souvent de petites tailles (figure 3.7). Or on a fait l'hypothèse que le robot se trouvait déjà en situation d'interaction. Il est donc normal de supposer qu'il se trouve à une distance raisonnable des différents membres de l'interaction et qu'une détection large de quelques pixels uniquement est probablement une fausse détection. Chaque fois qu'un visage est détecté, les pixels qui le composent sont ajoutés à la base d'entraînement avec l'étiquette c. De manière similaire, on souhaite sélectionner des pixels qui ne sont pas de la peau. Pour sélectionner de tels pixels, on peut choisir les pixels se trouvant sur les bords haut et bas de l'image, car il est moins probable que ces pixels contiennent de la peau. Ces pixels sont ajoutés à la base d'entraînement, avec l'étiquette $\bar{c}$.

Il est important de noter que l'information de positions dans l'image des pixels ajoutés à la base n'est pas importante. Seules les informations de couleur (R,V,B) et l'étiquette des pixels ajoutés seront prises en compte pour l'apprentissage des modèles capteurs.

Comme on l'a dit, un changement d'illumination de la scène soudain pourrait amener le besoin d'apprendre à nouveau les modèles capteurs. Pour éviter que les performances du détecteur de peau ne décroissent au cours du temps, la base de pixels est remise à jour à intervalles de temps régulier, en ajoutant de nouveaux pixels étiquetés grâce à l'algorithme de [VJ01]. Le processus complet de création de la base d'apprentissage est résumé dans l'algorithme 1.

Algorithme 1 : Processus de création de la base de pixels de peau

```
Data : Images du flux vidéo
Result : Base de pixels de peau étiquetés
while Nombre de pixels de visages insuffisant do
    Lire l'image
    Égaliser l'histogramme
    Détecter les visages à l'aide de [VJ01]
    foreach Pixel (R,V,B) détecté do
        Ajouter (R,V,B,c) à la base
    end
end
```

Il peut s'avérer que l'algorithme de Viola & Jones ne détecte pas de visage au cours des première trames (mauvaise orientation du visage, mauvaise position du robot). Il est donc raisonnable d'apprendre des modèles capteurs "par défaut" avec des pixels de peau enregistrés

en mémoire précédemment par le robot.

3.3.2.3 Apprentissage des modèles capteurs

On souhaite donc apprendre les modèles capteurs P(R|c), P(V|c), P(B|c), P(R|$\bar{c}$), P(V|$\bar{c}$), et P(B|$\bar{c}$). On a pour cela à notre disposition une base de pixels dont on connaît la couleur (R,V,B) et l'étiquette (c ou $\bar{c}$). Il est donc possible pour les deux étiquettes de transformer l'histogramme des pixels de la base en distribution de probabilités connaissant l'étiquette, soit P(R|C) (resp. P(V|C), P(B|C)).

On appelle $H_{c,R}(i)$ (resp. $H_{c,V}(i)$, $H_{c,B}(i)$), i ∈ [0,255] l'histogramme calculé sur les échantillons positifs de la base d'entraînement, sur les valeurs rouges (resp. vertes, bleues). De manière similaire, on appelle $H_{\bar{c},R}(i)$ (resp. $H_{\bar{c},V}(i)$, $H_{\bar{c},B}(i)$), i ∈ [0,255] l'histogramme calculé sur les échantillons négatifs de la base d'entraînement, sur les valeurs rouges (resp. vertes, bleues). Les vraisemblances P(R|c) et P(R| $\bar{c}$) sont obtenues de la manière suivante :

$$P(R|c) = \frac{H_{c,R}(R)}{\sum_{i=0}^{255} H_{c,R}(i)} \tag{3.19}$$

$$P(R|\bar{c}) = \frac{H_{\bar{c},R}(R)}{\sum_{i=0}^{255} H_{\bar{c},R}(i)} \tag{3.20}$$

Le procédé de création des modèles capteurs est résumé dans l'algorithme 2.

Algorithme 2 : Apprentissage des modèles capteurs

Data : Base de pixels (R,V,B,C)
Result : Modèles capteurs : P(R|C), P(V|C), P(B|C)
foreach *C in {c,$\bar{c}$}* **do**
  Calcul de $H_{C,R}$,$H_{C,V}$,$H_{C,B}$
  Normalisation en modèles capteurs
end

Pour finir l'entraînement du classifieur, on calcule l'a priori sur l'appartenance du pixel considéré aux échantillons positifs (resp. négatifs) :

$$P(c) = \frac{N_c}{N_c + N_{\bar{c}}} \tag{3.21}$$

$$P(\bar{c}) = \frac{N_{\bar{c}}}{N_c + N_{\bar{c}}} \tag{3.22}$$

Où N_c (resp. $N_{\bar{c}}$) est le nombre d'échantillons positifs (resp. négatifs) de la base d'entraînement. Cet a priori est donné par les proportions d'échantillons positifs et négatifs de la base

d'entraînement. Si on souhaite rendre l'a priori le moins informatif possible (ne connaissant pas réellement la probabilité qu'un pixel appartienne à la classe peau dans l'image), on peut fixer $P(c) = P(\bar{c}) = 0.5$.

3.3.2.4 Passage des pixels détectés au visage

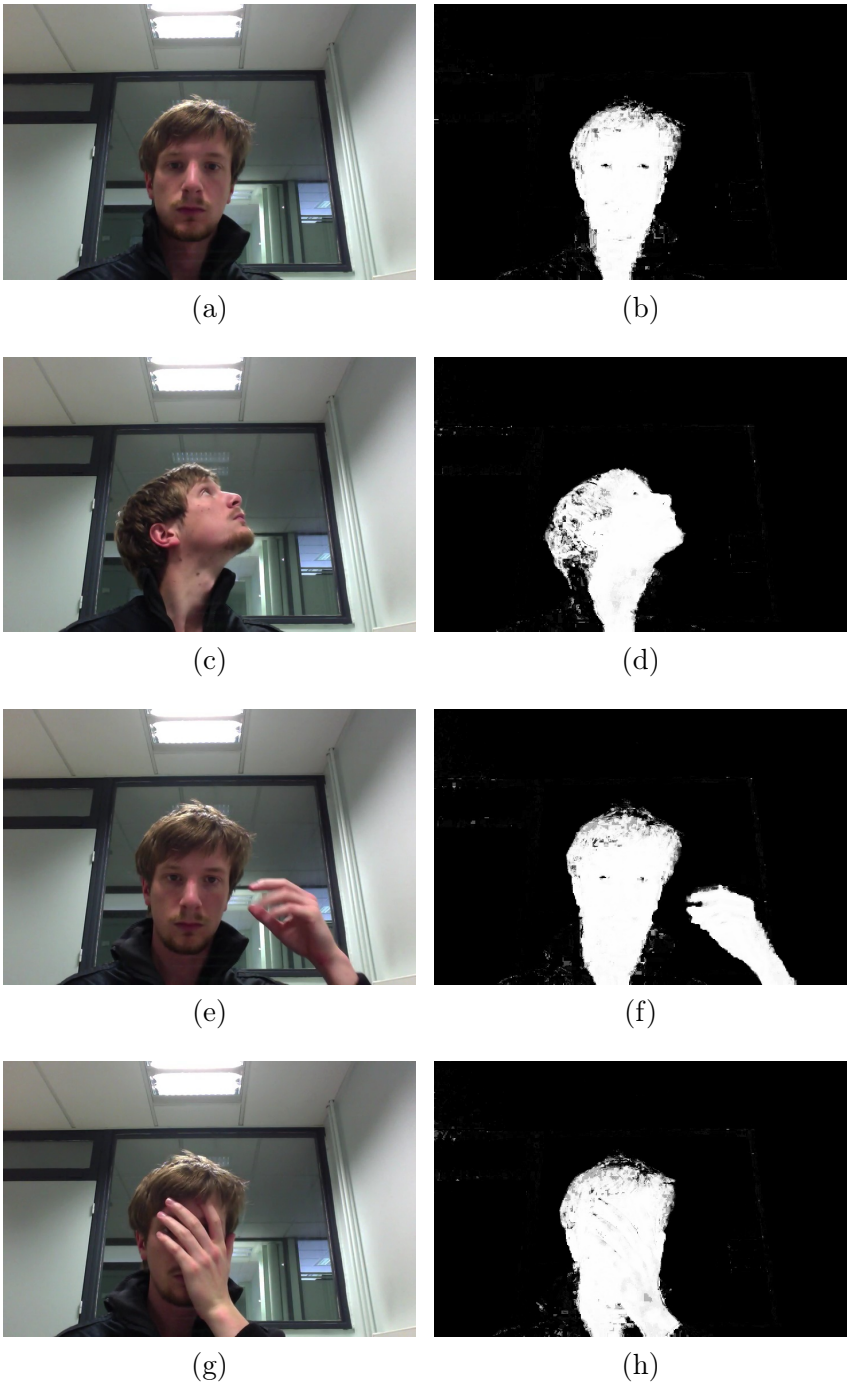
Comme on l'a dit, utiliser la détection de peau dans l'image permet d'éviter la détection de caractéristiques morphologiques, qui peut être compliquée dans le cas de capteurs de faible qualité. Cependant, en procédant ainsi, la géométrie et le contenu sémantique de l'image sont perdus. L'unique information dont on dispose au terme de la détection de peau est une information pixel par pixel d'appartenance ou non à la catégorie "peau". Il est donc nécessaire de reconstruire des visages à partir des pixels de peau détectés.

Pour obtenir des propositions de visages dans la vidéo à partir des pixels détectés, on considère les zones détectées connexes. Toute zone ayant une aire inférieure à un seuil arbitraire donné est supprimée. Ceci permet de limiter le nombre de fausses détections. On calcule le centre de gravité des zones restantes, et on obtient ainsi des pixels candidats pour une position de visage dans la vidéo.

Quelques résultats de la classification de peaux sont visibles dans la figure 3.9. Le classifieur est dans ce cas là entraîné sur deux détections de visage successives données par l'algorithme de Viola & Jones. Le temps moyen de détection par trame est 10 fois inférieur à celui de Viola & Jones, ce qui rend l'algorithme viable pour une analyse de la vidéo en temps réel. On peut voir que les cheveux sont aussi détectés, ce qui ne pose pas de problème. Ceci est dû au fait que la zone procurée par l'algorithme de [VJ01] contient des cheveux, dont les pixels sont appris. Il est possible de remédier à un tel problème en découpant plus précisément le visage obtenu, soit directement, soit en utilisant les classifieurs d'yeux et de bouche entraînés par l'algorithme de [VJ01]. En obtenant les informations de position des yeux et de la bouche de le visage, cela permettra de sélectionner plus précisément les pixels appartenant clairement à la peau. Cependant la détection des cheveux des personnes n'a jamais été un problème dans les expériences menées.

On note que le visage est toujours détecté, même dans le cas d'une occlusion par la main (figure 3.9h), ce qui représente un avantage car les occlusions partielles par la main peuvent arriver de manière courante au cours d'une interaction. Toutefois, la détection n'est pas parfaite : si la main ou toute autre partie dénudée du corps de la personne se trouve dans le champ de vision du robot, elle sera détectée. Il est logique que la détection de pixels de peau augmente le taux de détection par rapport à la détection de visages, car elle couvre un domaine plus large. Ceci augmente donc le nombre de fausses alarmes (pixels détectés n'appartenant pas à un visage d'être humain). Ça ne représente pas un problème :

— On retire les blocs de pixels détectés qui sont trop petits pour représenter un visage à l'aide d'opérations morphologiques classiques, ce qui permet de retirer un bon nombre de pixels détectés n'appartenant pas à un visage



(a) (b) (c) (d) (e) (f) (g) (h)

FIGURE 3.9 – Quelques exemples de détection de peau à l'aide de la fusion probabiliste instantanée, dont les modèles capteurs ont été appris sur 2 détections de visage successives.

— L'information vidéo sera fusionnée avec l'information audio, ce qui éliminera les fausses détections dans une majorité des cas.
— La majorité du temps, la tête d'une personne se trouve au-dessus de toutes les parties de son corps, dénudées ou non. Dans le cas où un doute subsiste quant à l'origine de la parole, il semble naturel de choisir la zone de peau la plus haute sur l'image.

Cette méthode possède plusieurs avantages : elle est efficace avec un petit nombre d'éléments dans la base d'apprentissage (une seul prise du visage peut suffire pour la détection), et elle est adaptable, du fait de la création de la base d'apprentissage directement en ligne. La base d'apprentissage peut être constamment remise à jour (par exemple pour faire face à un changement soudain et fort d'illumination) en utilisant sur l'espace de quelques trames l'algorithme de Viola & Jones. Finalement, comme cette détection ne repose pas sur des caractéristiques morphologiques de l'image, elle fonctionnera avec des images de faible résolution, ce qui augmente encore son intérêt pour l'économie des ressources de calcul.

3.3.3 Localisation de la source sonore

A ce stade, on dispose de candidats pour la position d'un visage dans la vidéo. Cependant, les caractéristiques extraites de la vidéo dans ce travail ne donnent pas d'information sur la personne en train de parler. Pour cela, on choisit de localiser la source sonore dans le flux audio, grâce à un procédé reposant sur la différence de temps d'arrivée du son entre les deux microphones. Ce procédé est résumé dans la figure 3.10.

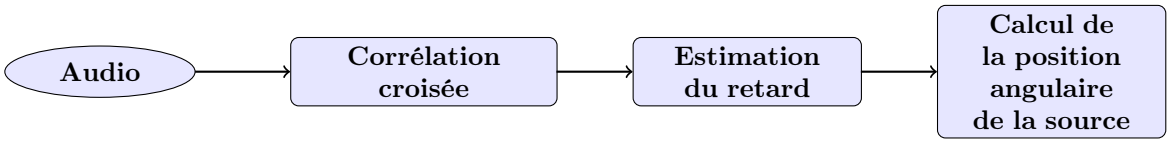


FIGURE 3.10 – Procédé de localisation de la source sonore

Le problème de la localisation d'une source audio à l'aide d'une paire de microphones est un problème connu qui peut se présenter de la manière suivante (figure 3.11) : une source de son "infiniment" distante S émet un signal s(n) échantillonné à l'acquisition par deux microphones M1 et M2.

On dénomme x_1(n) et x_2(n) les signaux discrets acquis respectivement par M1 et M2. Ces signaux peuvent donc s'exprimer en première approximation comme :

$$x_1(n) = s(n) + n_1(n) \tag{3.23}$$
$$x_2(n) = s(n - \Delta N) + n_2(n) \tag{3.24}$$

où ΔN est le délai d'arrivée du signal entre les microphones M1 et M2 et n_i est le bruit associé aux microphones. L'objectif ici est de calculer l'angle $\theta = \widehat{M_1 M_2 S}$, connaissant la distance D entre les deux microphones. Le moyen le plus efficace d'obtenir cet angle est de

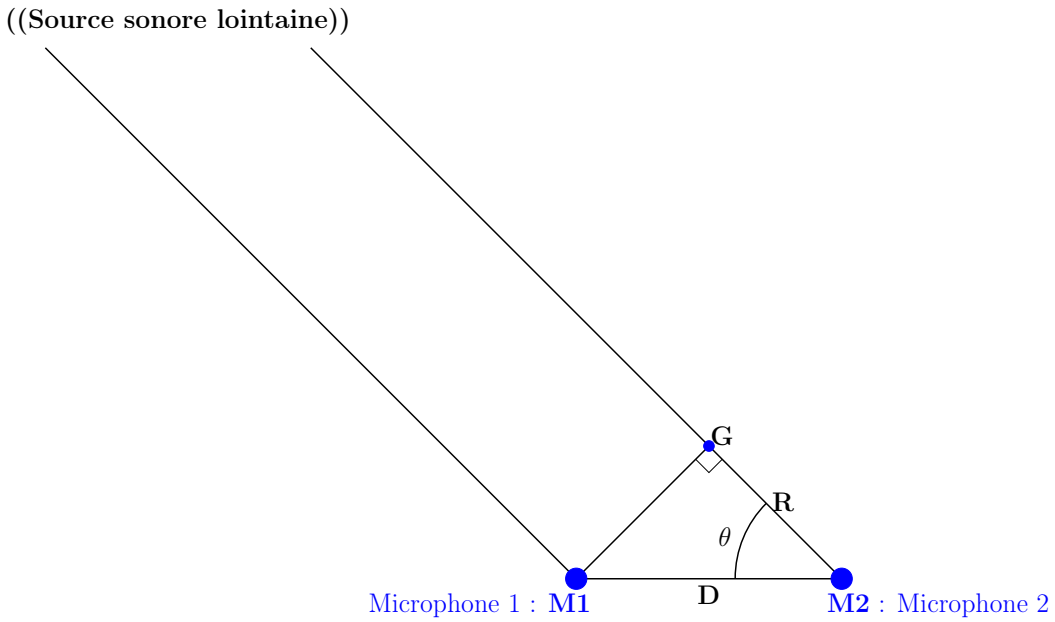


FIGURE 3.11 – Problème de la localisation d'une source de son

calculer la différence de temps d'arrivée du signal entre les deux microphones M1 et M2 (Time Difference of Arrival, TDOA). La vitesse du son dans l'air étant connue, il est possible de calculer la distance R parcourue par le son pendant le TDOA. Prenant en compte l'hypothèse de distance infinie de la source, il est possible de calculer l'angle θ :

$$\theta = acos\left(\frac{R}{D}\right) = acos\left(\frac{TDOA \cdot v}{D}\right) = acos\left(\frac{\Delta N \cdot v}{D \cdot Fe}\right) \tag{3.25}$$

où v est la vitesse du son dans l'air, et F_e est la fréquence d'acquisition du signal. Il est donc nécessaire de calculer le retard en nombre d'échantillons entre les deux microphones ΔN. Pour ce faire, la méthode de la corrélation croisée est la plus simple. La corrélation croisée $Corr$ entre deux signaux s'écrit de la manière suivante :

$$Corr(\tau) = \mathbb{E}\left[x_1(n)\, x_2(n-\tau)\right] \tag{3.26}$$

où $\mathbb{E}$ représente l'espérance mathématique.

Le retard en nombre d'échantillons ΔN est donné par l'argmax de la corrélation :

$$\Delta N = \underset{\tau}{\text{argmax}}\left(Corr(\tau)\right) \tag{3.27}$$

Cette modélisation possède quelques inconvénients :

— Elle est incapable de localiser précisément la source dans l'espace autour des microphones. Elle nécessite de supposer que la source émettrice se trouve à l'infini. Elle

permet donc d'obtenir une position angulaire en azimut de la source par rapport aux deux microphones mais ne peut pas présager de la distance réelle de la source.
— La supposition d'une source émettrice placée à l'infini peut poser problème. Bien sûr dans la réalité, cela n'est pas vérifié, mais il est tout de même nécessaire d'avoir $||SM_1|| \approx ||SM_2|| \gg ||M_1M_2||$. Avec des microphones espacés d'une quinzaine de centimètres, cette hypothèse n'est pas toujours vérifiée, et posera problème si la source se trouve très excentrée par rapport aux microphones (très à gauche ou très à droite), ou trop proche des microphones. Dans ce cas, la distance R devient plus grande que D, ce qui rend le calcul du retard impossible.
— Ce modèle ne prend pas en compte les trajets secondaires du son. Seul le trajet direct est considéré. Ceci n'est pas un problème en première approximation, car on considère que le robot est en position d'interaction, et donc que l'énergie apportée par le trajet direct (a priori sans obstacles) du son sera dominante par rapport aux autres trajets.

Ainsi le son est traité en 2 étapes, de la manière suivante (algorithme 3) :

— Une fenêtre glissante est appliquée sur le flux audio, en synchronisation avec la vidéo, et un filtre de Butterworth coupe-bas est appliqué sur chacune des fenêtres pour atténuer le bruit inhérent aux microphones et au robot.
— La corrélation est calculée et la position angulaire de la source par rapport aux microphones en est déduite grâce à la méthode ci-dessus.

Algorithme 3 : Processus de localisation de source sonore

Data : Flux audio
Result : Position angulaire de la source par rapport aux microphones
foreach *Fenêtre temporelle* **do**
    Filtrage de Butterworth
    Calcul de la corrélation
    Calcul du retard entre les deux microphones
    Déduction de la position angulaire de la source sonore
end

Cet algorithme a été testé à l'aide de deux microphones espacés de 15 centimètres : une source émettrice de sons est enregistrée par les deux microphones, en se déplaçant autour de ces microphones (figure 3.12), toujours à une distance de 1m, dans 16 positions angulaires différentes. Dans le cas d'une distance de 1m, l'hypothèse de la source se trouvant à une distance infinie de la paire de microphones n'est pas respectée. Cependant, ce sont les conditions réalistes d'une interaction.

Deux types de sons ont été testés : un son percussif, où la source claquait dans ses mains, et le cas où la source émettait des syllabes courtes. La comparaison entre angle réel et mesuré est disponible dans les tableaux 3.1 et 3.2. Comme on l'avait prédit, les angles se trouvant sur les côtés du robot ne peuvent pas être calculés. On note que plus on se rapproche de 90 degrés (en face des deux microphones), plus la méthode fournit une estimation précise de l'angle. L'erreur quadratique moyenne est consistante entre les deux tests : 8.1 degrés pour le tableau

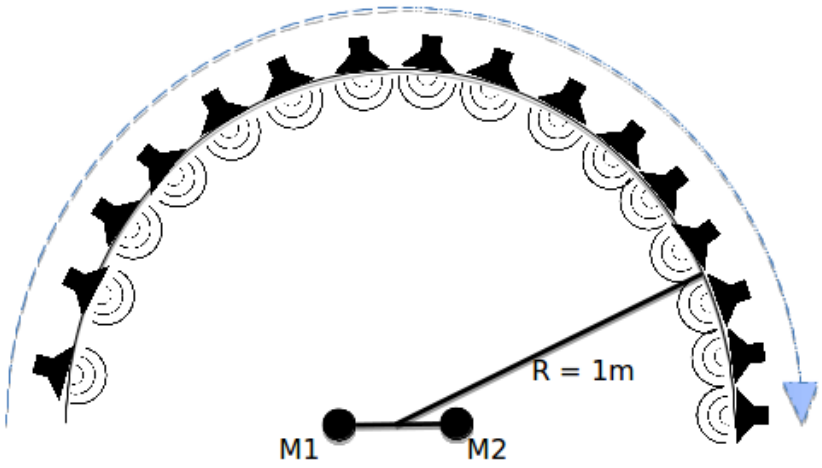


FIGURE 3.12 – Test de la corrélation croisée pour la localisation de source sonore

3.1 et 8.3 pour le tableau 3.2 (les angles impossibles à mesurer ne sont pas pris en compte dans cette situation). Si les deux premières mesures (sur la gauche des deux microphones) ne sont pas prises en compte, cette erreur tombe à 6 degrés.

Ces valeurs sont acceptables : en effet, si deux personnes sont séparées par un angle inférieur à 8 degrés, elle devront être éloignées du robot pour être différentiables à la détection de peau. Au cours d'une interaction, les êtres humains sont rarement extrêmement proches les uns des autres.

3.3.4 Fusion audiovisuelle à l'aide d'un filtre temporel probabiliste

A ce stade, on dispose d'informations sur chacune des modalités :
— Sur chaque trame de la vidéo, des pixels peuvent être détectés (ou non), donnant de possibles candidats pour une détection de visage ;
— Sur le flux audio, à chaque instant d'observation, une source audio peut être localisée (ou non) en azimut par rapport au robot ;

On souhaite fusionner ces deux informations à chaque instant pour détecter le locuteur dans la conversation, en prenant en compte l'information passée. Tout d'abord, il est nécessaire de définir précisément l'espace de perception du robot. Celui-ci est illustré dans la figure 3.13.

Comme on peut le voir dans cette figure, les champs de perception angulaire des deux capteurs sont différents : on nomme β le champ angulaire de vision. On considère que le champ angulaire de perception audio est de 180°, soit l'espace se trouvant devant le robot. Finalement, on appelle α l'angle auquel la caméra est tournée. L'étendue spatiale de perception audio étant supérieure à celle de la perception vidéo, il semble naturel pour fusionner l'information apportée par les deux capteurs d'utiliser la localisation angulaire.

Angle réel (degrés)	Angle mesuré (degrés)
0	NaN
12	NaN
26	10
40	26
56	46
72	60
80	73
91	85
101	96
110	105
120	114
127	124
140	135
149	155
161	NaN
180	NaN

TABLE 3.1 – Sensibilité de la corrélation croisée : une source claquant de ses mains

Angle réel (degrés)	Angle mesuré (degrés)
0	NaN
12	NaN
26	NaN
40	25
56	50
72	60
80	73
91	82
101	93
110	108
120	114
127	117
140	139
149	155
161	NaN
180	NaN

TABLE 3.2 – Sensibilité de la corrélation croisée : une source émettant des syllabes courtes

Pour ce faire, on doit pouvoir transformer une observation vidéo en position angulaire par rapport au robot, soit la position d'un pixel (x,y) dans l'image en angle γ. On propose une transformation simple :

$$\gamma = \alpha + x \cdot \frac{W}{\beta} \tag{3.28}$$

où

— x est la position du pixel dans la largeur de l'image fournie par la vidéo

— W est la largeur de l'image fournie par la vidéo : ainsi $\frac{W}{\beta}$ représente le nombre de pixels par degré dans la dimension de largeur.

Pour transformer la position d'un pixel en angle, on ne peut s'intéresser qu'à la position x du pixel dans la dimension de la largeur de l'image : en effet, pour l'audio l'angle est donné en azimut (soit la direction de la largeur de l'image), la position verticale de la source sonore n'est pas calculable. Cette transformation n'est pas exacte, car toute caméra induit des déformations, qu'il est nécessaire de prendre en compte si on souhaite une très forte précision. Toutefois, cette approximation est suffisante en première approche, si on considère que le robot sera proche des personnes en interaction.

Ainsi on obtient à chaque instant t d'observation :

— Un vecteur d'observations vidéos $z_t^v = \{z_t^1, \cdots, z_t^N\}$ correspondant aux positions angulaires des pixels candidats pour des visages

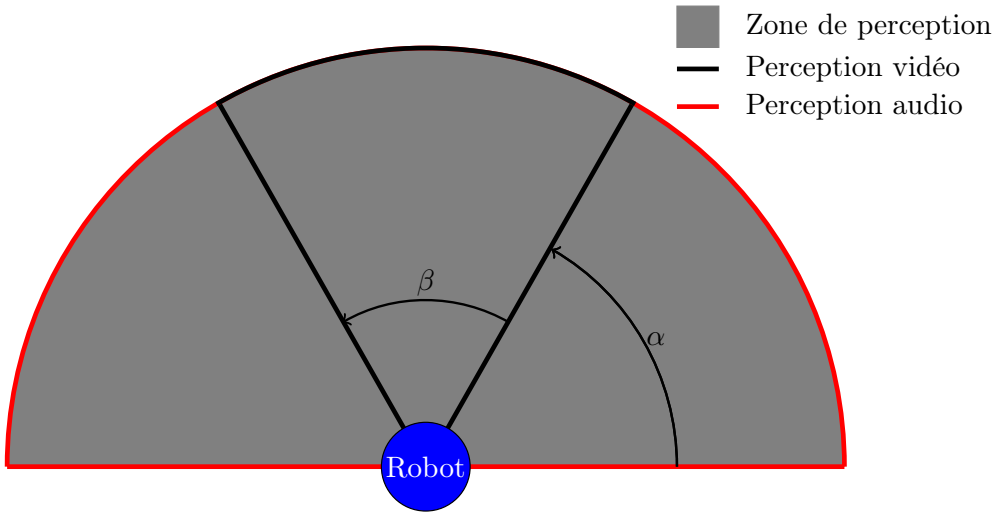


FIGURE 3.13 – Illustration de l'espace de perception du robot

— Une observation audio z_t^a correspondant à la position angulaire de la source audio détectée (dans le cas d'une détection)

Pour fusionner ces deux valeurs dans le temps, on propose d'utiliser une instanciation du filtre temporel probabiliste décrit en section 3.2.2.1. Pour ce faire il est nécessaire de diviser l'espace de perception du robot en espace d'états discret. On cherche à déterminer la position angulaire du locuteur dans une conversation : comme la précision angulaire de la détection audio est faible, et que deux personnes en conversation seront rarement très proches, on choisit de diviser l'espace angulaire en 18 états discrets possibles, chacun composé d'une zone angulaire de 10 ° (figure 3.14).

On dispose ainsi d'un espace d'états $\mathcal{X} = \{\mathcal{X}^0, \cdots, \mathcal{X}^{17}\}$ de taille 18 correspondant aux positions angulaires possibles de la source par rapport au robot.

De la même manière que le filtre présenté en section 3.2.2.1, le filtre s'exécute ici en deux parties classiques : prédiction et correction.

3.3.4.1 Réalisation du filtre - Prédiction

La prédiction est la version discrète de celle présentée dans l'équation 3.8 :

$$\underbrace{p(x_t|z_{t-1}^a, z_{t-1}^v)}_{\text{Prédiction}} = \sum_{x_{t-1}} \left(\underbrace{p(x_t|x_{t-1})}_{\text{Modèle dynamique}} \cdot \underbrace{p(x_{t-1}|z_{t-1}^a, z_{t-1}^v)}_{\text{Estimée à t-1}} \right) \tag{3.29}$$

Comme on l'a décrit en section 3.2.2.1, il est nécessaire de connaître le modèle dynamique du problème, soit la distribution de probabilité $p(x_t|x_{t-1})$.

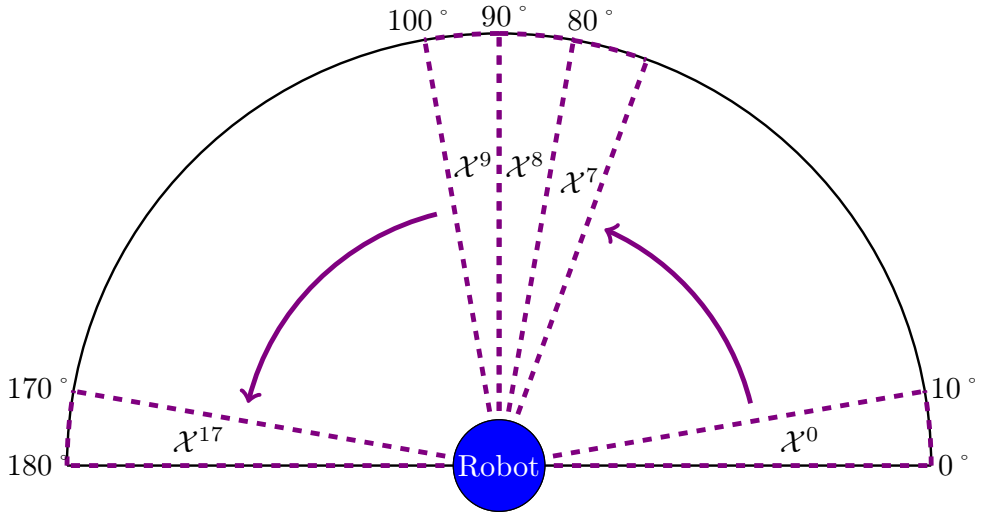


FIGURE 3.14 – Décomposition de l'espace de perception du robot en états discrets

Modèle dynamique du problème

Le modèle dynamique correspond à la probabilité d'évolution naturelle de l'état du locuteur dans la conversation, c'est à dire la probabilité que le locuteur se trouve à la position $x_t \in \mathcal{X}$ à l'instant t connaissant sa position x_{t-1} à l'instant t-1. C'est la distribution $\mathrm{p}(x_t|x_{t-1})$. Dans notre cas, cela revient a être capable de prédire un changement éventuel de position d'un locuteur, ce qui est difficile. Bien sûr, une personne est mobile lorsqu'elle parle, mais il est compliqué de prévoir un mouvement particulier a priori. On propose donc un modèle dynamique gaussien centré sur x_{t-1}.

$$p(x_t|x_{t-1}) = \frac{1}{\sigma\sqrt{2\pi}} \exp \frac{(x_t - x_{t-1})^2}{2\sigma^2} \tag{3.30}$$

où σ est l'écart-type de la loi, arbitrairement fixé. Ceci permet de prendre en compte un léger mouvement d'une source donnée, mais ne permet pas de prévoir un changement complet de locuteur (qui se trouverait dans un état complètement différent du locuteur précédent). Une illustration de cette loi pour le cas où l'état à t-1 et $\mathcal{X}^7$, soit la distribution de probabilités $\mathrm{p}(x_t|\mathcal{X}^7_{t-1})$ est montrée dans la figure 3.15.

3.3.4.2 Réalisation du filtre - Correction

La prédiction est ensuite corrigée, à l'aide des observations, comme dans l'équation 3.9. Toutefois, contrairement au cas de l'équation 3.9, on a ici deux sources d'observations, l'audio et la vidéo. Toutefois, de la même manière que dans le cas de la fusion instantanée, on fait l'hypothèse que ces deux sources sont conditionnellement indépendantes, ce qui permet de

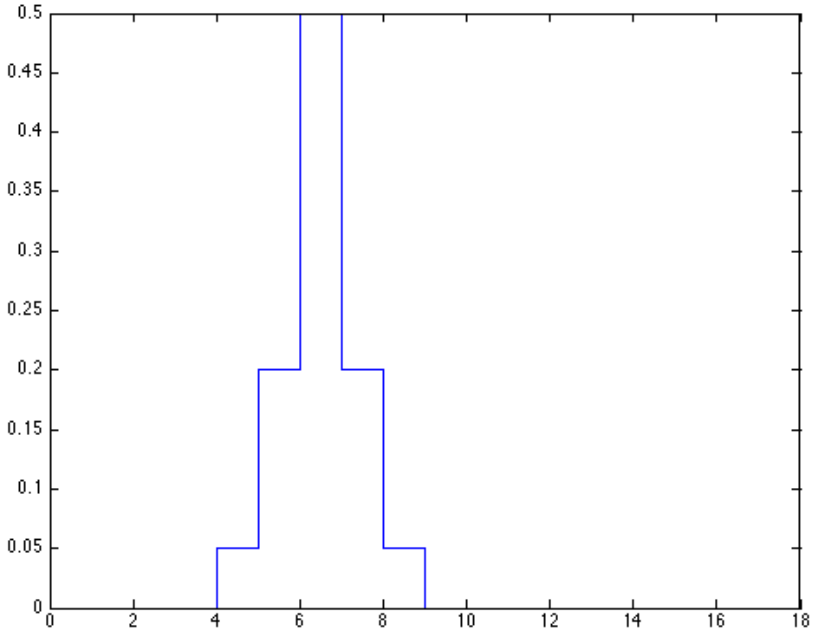


FIGURE 3.15 – Exemple de modèle dynamique : $p(x_t|\mathcal{X}_{t-1}^7)$: en abscisse, les différents états (18 états possibles) ; en ordonnées, la probabilité associée par le modèle dynamique.

multiplier les deux modèles capteurs, soit une étape de correction :

$$\underbrace{p(x_t|z_t^a, z_t^v)}_{\text{Estimée à t}} = \alpha \cdot \underbrace{p(z_t^a|x_t)}_{\text{Modèle capteur audio}} \cdot \underbrace{p(z_t^v|x_t)}_{\text{Modèle capteur vidéo}} \cdot \underbrace{\widehat{p}(x_t|z_t^a, z_t^v)}_{\text{Prédiction}} \tag{3.31}$$

Il est donc nécessaire de définir les deux modèles capteurs : audio et vidéo.

Modèle capteur audio

Le modèle capteur audio correspond à la probabilité d'observer une source audio z_t^a connaissant l'état du locuteur x_t à l'instant t, soit $p(z_t^a|x_t)$. La localisation de source fournit un angle $z_t^a \in [0; 180]$. Ici encore une gaussienne semble adaptée pour décrire le modèle capteur audio. On choisit donc une gaussienne centrée sur la valeur médiane des positions angulaires contenues dans l'état S_t. Un exemple d'une telle distribution est donné dans la figure 3.16. Dans cet exemple, on décrit la distribution $p(z_t^a|S_t^7)$, soit la probabilité de localisation d'une source sachant que le locuteur se trouve dans l'état S_7 (position comprise entre 70 ° et 80 °). On aura dans ce cas une gaussienne centrée sur la valeur angulaire 75 ° .

On choisira un écart-type large, reflétant l'erreur quadratique moyenne produite par les mesures données dans les tableaux 3.1 et 3.2.

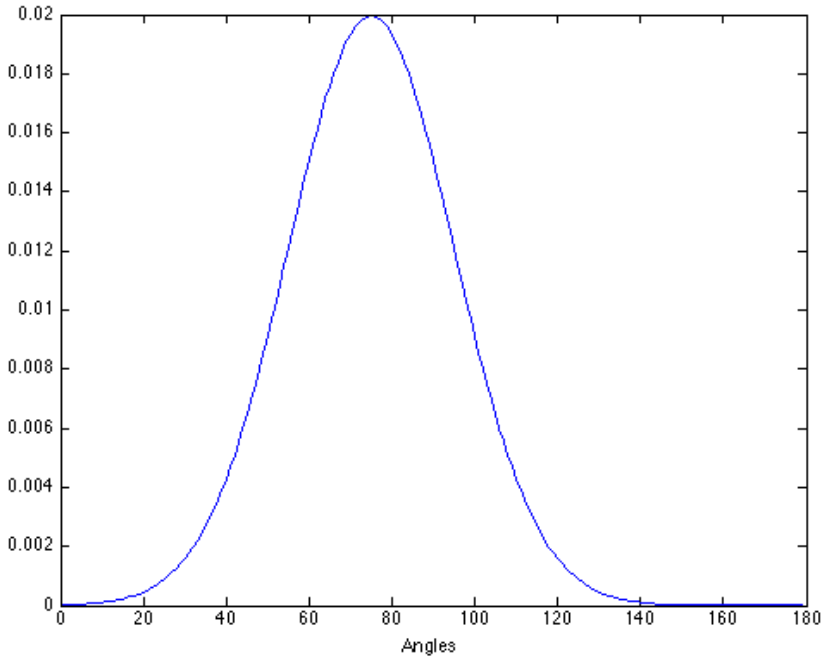


FIGURE 3.16 – Exemple de modèle capteur audio : $p(z_t^a|\mathcal{X}_t^7)$. En abscisse, les différents angles possibles d'observation d'une source sonore ; en ordonnée la probabilité associée par le modèle capteur

Modèle capteur vidéo

Le modèle capteur vidéo correspond à la probabilité d'observer une source à une position angulaire z_t^v à l'instant t connaissant la position angulaire du locuteur à l'instant t $x_t \in S$, soit $p(z_t^v|x_t)$ Ici, il est nécessaire de prendre en compte le champ de vision, qui est moindre que le champ de perception audio. Le modèle capteur vidéo est donc défini sur moins de positions angulaires que le modèle capteur audio. On se trouve ainsi face à deux cas possibles :

— **Si x_t se trouve en dehors du champ de vision :** le locuteur ne peut pas être visible par la caméra. Dans ce cas, il est possible de détecter des visages uniformément au sein de l'image. Cela correspond donc à une distribution uniforme sur l'ensemble des positions angulaires se trouvant dans le champ de vision.

— **Si x_t se trouve dans le champ de vision :** le locuteur est visible par la caméra, il est donc probable de détecter un visage à la position angulaire x_t. De la même manière que pour le modèle capteur audio, on utilise dans ce cas une distribution gaussienne centrée sur la valeur médiane des positions angulaires contenues dans l'état x_t (figure 3.16). La détection visuelle étant toutefois plus précise que la localisation de source, on pourra utiliser une gaussienne d'écart-type moindre que celui du modèle capteur audio.

3.3.4.3 Réalisation du filtre - Choix de l'état et de l'observation vidéo

Une fois la distribution de probabilités sur les états estimée, on choisit l'état maximisant la distribution a posteriori :

$$\mathcal{X}_{final} = \underset{x_t \in \mathcal{X}}{\operatorname{argmax}} \left(p(x_t | z_t^a, z_t^v) \right) \tag{3.32}$$

Une fois l'état $\mathcal{X}_{final}$ obtenu, on souhaite diriger l'attention du robot, dans la mesure du possible vers le visage détecté. Ici encore, on se trouve face à 2 cas possibles :

— Si $\mathcal{X}_{final}$ se trouve dans le champ de vision du robot, on choisit l'observation vidéo la plus proche de cet état :

$$z_t^{final} = \underset{z_t^i}{\operatorname{argmin}} \left| z_t^i - \mathcal{X}_{final} \right| \tag{3.33}$$

Si aucune observation vidéo n'a été faite (aucun pixel de peau détecté), c'est l'observation audio z_t^a qui est considérée.

— Si $\mathcal{X}_{final}$ ne se trouve pas dans le champ de vision du robot, on ne peut pas choisir d'observation vidéo à l'instant t, et seule cette valeur est gardée.

Au terme de l'exécution de ce filtre, on obtient donc un état qui peut devenir le centre de l'attention du robot. Ceci permet au robot, à l'aide d'un asservissement simple [Orc14], de tourner la tête pour mettre le visage détecté au centre de son champ de vision.

3.4 Résultats

3.4.1 Présentation du robot Reeti

La plateforme sur laquelle a été réalisée cette étude est Reeti, petit robot "humanoïde" produit par Robopec [1] (figure 3.17).

Reeti dispose d'une tête mobile, contenant des caméras RGB pouvant délivrer une image 640x480. Il peut tourner la tête horizontalement, latéralement et verticalement. Il dispose aussi de deux microphones à sa base permettant d'avoir accès au flux audio.

3.4.2 Résultats expérimentaux

Pour tester le système en situation réelle, nous avons demandé à des sujets de s'asseoir face au robot et de simuler une conversation. Les microphones de reeti étant d'une qualité médiocre, pour simuler la conversation, nous avons demandé aux sujets d'émettre du son en continu lors de leur prise de parole. Différents scénarios ont été explorés, avec des personnes

1. http ://www.reeti.fr/index.php/en/

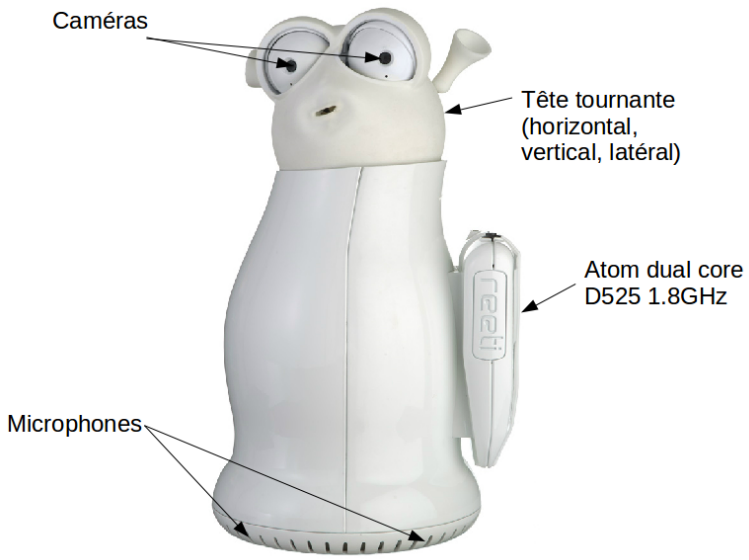


FIGURE 3.17 – Le robot Reeti

pouvant entrer à tous moment dans la scène (parlantes ou non), plus ou moins proches du robot. Quelques vidéos sont disponibles à l'adresse suivante : Lien vers les vidéos.

On propose d'illustrer les résultats expérimentaux dans cette section par deux exemples concrets : un cas simple avec deux humains immobiles en conversation, et un cas plus complexe avec un humain mobile et un autre immobile.

3.4.2.1 Expérience 1 : conversation entre deux locuteurs immobiles

Dans cette première expérience, les deux locuteurs sont assis face au robot (figure 3.18) : chacun des deux humains parle successivement, sans concertation préalable.

FIGURE 3.18 – Expérience 1 : deux humains en conversation face au robot

Une illustration de la chaîne de traitements complète est donnée dans la figure 3.19. Dans ce cas, la personne en train de parler (humain de gauche sur l'image) est visible par le robot. Comme on peut le voir (figure 3.19 (b)), la localisation sonore manque de précision. Ceci est dû à deux éléments :

— L'algorithme de localisation de source sonore employé n'est pas robuste au bruit produit par le robot, bien que celui-ci soit pris en compte par le filtrage de Butterworth.
— Le replacement de l'angle localisé dans l'image n'est pas parfait : comme on l'a décrit dans la section 3.3.4, on utilise une transformation linéaire, dans le but d'économiser du temps de calcul. Or, toute caméra induit une déformation dans l'image. Plus l'angle détecté se trouve sur le côté du robot, moins sa position à l'image sera précise.

La détection de peau, illustrée par la figure 3.19 (c) montre bien les deux visages des humains présents dans la conversation, et l'attention du robot (cercle bleu dans la figure 3.19 (d)) est bien portée sur le visage du locuteur actuel de la conversation.

La fusion permet de prendre facilement en compte un changement de locuteur, comme illustré par la figure 3.20. Lorsqu'un locuteur cesse de parler, il continue d'être détecté comme la dernière cible d'attention du robot : ceci est dû au modèle dynamique utilisé par le système. Sans information supplémentaire de la part de l'audio ou la vidéo, on considère que le locuteur n'a pas changé de place. Ainsi l'attention du robot reste focalisée sur la dernière personne ayant parlé (figure 3.20 (a) et (b)). A partir du moment où le locuteur change réellement, quelques trames sont nécessaires pour prendre en compte le changement. Ceci est dû au conflit entre la prédiction (le locuteur restera dans la même zone) et les nouvelles observations audio (le locuteur à changé de position).

3.4.2.2 Expérience 2 : conversation entre une personne immobile et une personne mobile

Le déroulement du scénario est le suivant :
— Au début du scénario, un premier humain est seul, immobile, visible par le robot, en train de parler. Cet humain restera immobile tout au long du scénario
— Un deuxième humain entre et sort de la scène de manière intermittente, en parlant successivement à l'intérieur et en dehors du champ de vision

L'objectif ici est de tester les performances du système lors du mouvement d'un des locuteurs. Pour illustrer l'expérience, on montre le scénario à 4 instants d'observation différents dans la figure 3.21. Sur cette figure, on peut voir la détection vidéo, la localisation audio, et le résultat de la fusion (cible d'intérêt détectée par le robot).

— **t = 23 : seul l'humain visible est locuteur.** Jusqu'à ce temps d'observation, seul l'humain présent dans la vidéo a parlé. La source sonore, placée idéalement face au robot, est bien localisée, et la détection de peau permet d'attirer l'attention du robot sur le visage du premier humain.
— **t = 49 : le deuxième humain entre en parlant dans le champ de vision du robot.** Dès lors que l'humain entre dans le champ de vision, il est visuellement

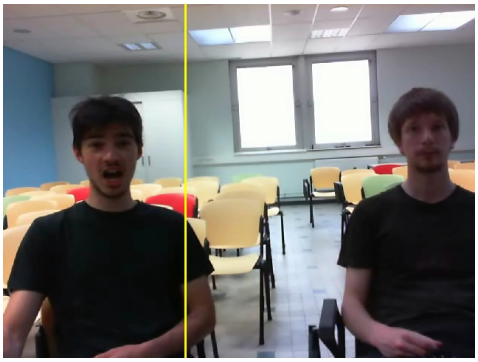
détecté par l'algorithme de détection de peau (figure 3.21 (e)). De la même manière la localisation de source sonore tends vers le deuxième humain (figure 3.21 (d)). La fusion audiovisuelle donne donc de manière juste l'humain 2 comme locuteur actuel.

— **t = 62 : l'humain 2 est en train de traverser le champ de vision du robot en parlant.** Une vitesse de déplacement volontairement grande a été choisie. Dans le scénario, l'humain traverse le champ de vision du robot en moins d'une seconde. Comme on peut le voir, la fusion temporelle n'est pas capable de suivre l'humain 2 : ceci est dû au manque de précision de la localisation sonore (figure 3.21 (g)). La localisation ayant un léger retard sur la vidéo, c'est l'humain 1 qui est localisé, alors qu'il est silencieux. Ainsi, lors de la fusion c'est l'humain 1 qui est détecté comme locuteur. Tout au cours du passage de l'humain dans le champ de vision du robot, la localisation sonore oscille entre la position de l'humain et celle de l'humain 2. Ceci est dû aux trajets secondaires et à la réverbération du son. Le détecteur audio n'étant pas assez robuste, la fusion temporelle ne peut pas compenser la perte constante de localisation du locuteur.

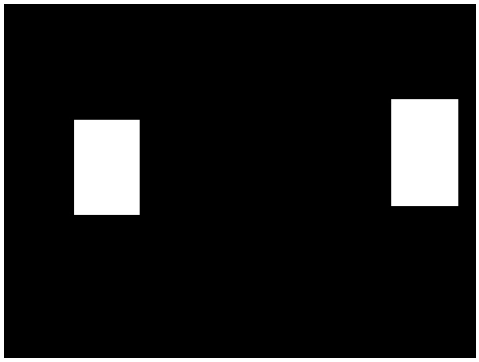
— **t = 84 : l'humain 2 s'est immobilisé hors du champ de vision du robot et continue de parler.** Cette fois, le locuteur n'est pas visible. Une fois l'humain 2 arrêté et parlant depuis quelques trames, la localisation sonore se stabilise autour de sa position (figure 3.21 (j)) : le trait vertical jaune signifie que la source de son entendue se trouve hors champ). La fusion détecte ainsi bien le locuteur en dehors du champ de vision du robot sur sa gauche (figure 3.21 (l)).

Le système réagit de manière correcte lorsque les mouvements du locuteur ne sont pas trop rapide, avec une vitesse de réaction de l'ordre d'une douzaine de trames, soit environ 0.5 secondes, ce qui est acceptable pour suivre une conversation naturelle entre deux humains. Dans le cas où la position du locuteur varie trop rapidement, le détecteur audio utilisé n'étant pas très précis, la fusion temporelle n'est pas capable de compenser pour traquer sa position. Lorsque le locuteur se trouve en dehors du champ de vision du robot, le système le détecte de manière précise.

(a) Image originale vidéo



(b) Détection de la source sonore (barre verticale)



(c) Détection de visages

(d) Fusion de l'information et détection finale (Cercle bleu)

FIGURE 3.19 – Illustration de la chaîne dans le cas où le locuteur est visible par le robot

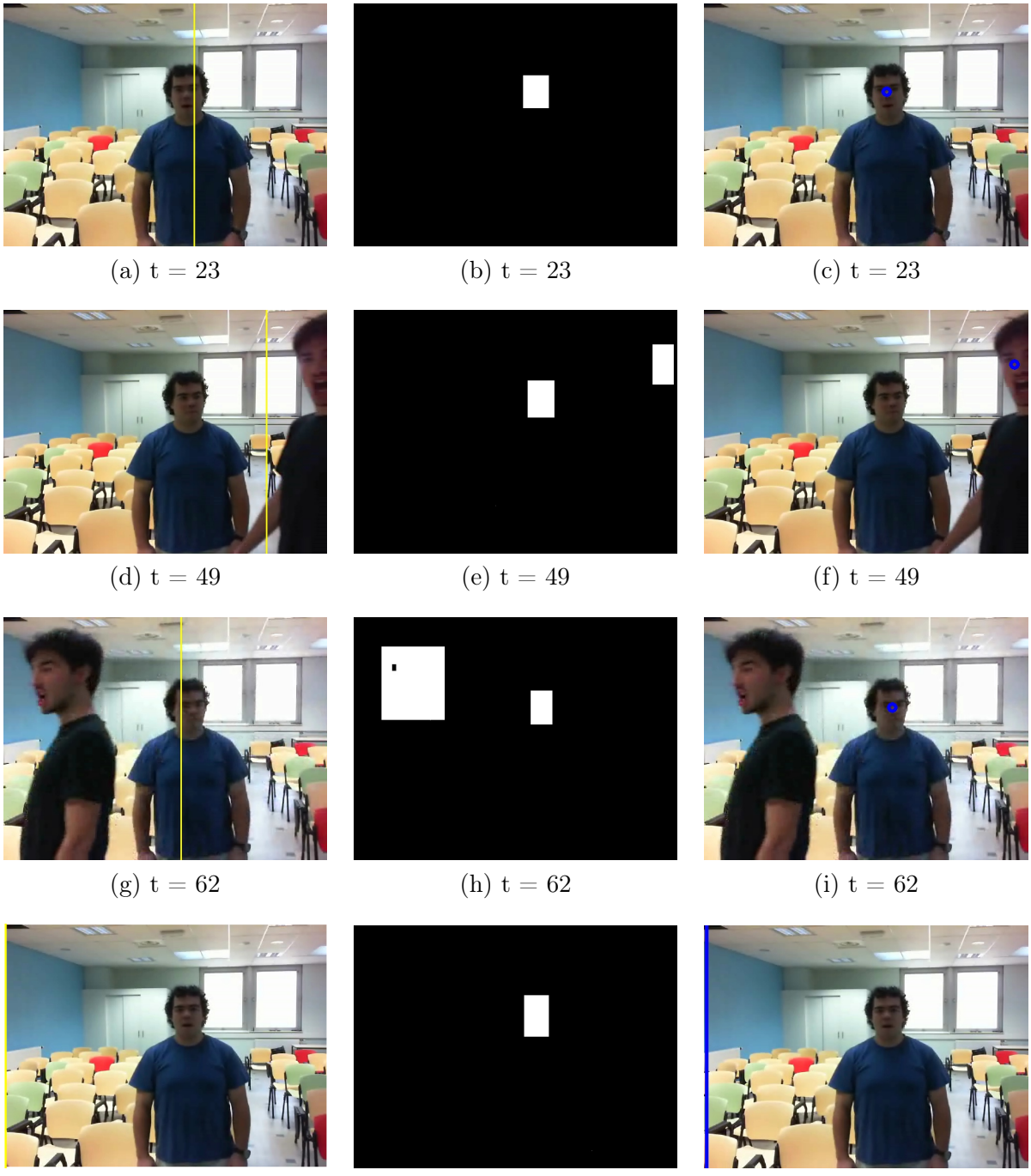
(a) Instant t : l'humain de droite cesse de parler

(b) Instant t+5 : l'humain de gauche commence à parler

(c) Instant t+10 : l'humain de gauche est détecté comme locuteur

FIGURE 3.20 – Changement de locuteur : lorsqu'un humain cesse de parler, le nouveau locuteur est détecté au bout d'environ 5 trames

Localisation audio	Détection vidéo	Résultat fusion



(a) t = 23 (b) t = 23 (c) t = 23

(d) t = 49 (e) t = 49 (f) t = 49

(g) t = 62 (h) t = 62 (i) t = 62

(j) t = 84 (k) t = 84 (l) t = 84

FIGURE 3.21 – Expérience 2 : fusion audiovisuelle au cours du temps

3.5 Conclusion

Dans ce chapitre, un système de détection de locuteurs successifs dans une conversation a été proposé. Il repose sur la fusion de données : la localisation angulaire de source sonore sur le flux audio et la détection de visages sur la vidéo. Le système fonctionne en temps réel, implémenté sur un petit robot. Il est robuste à certaines formes d'occlusion classiques d'une conversation (occlusion partielle ou totale du visage par la main), et détecte le locuteur de manière précise. Un changement de locuteur est pris en compte par le système, ainsi qu'un éventuel locuteur se trouvant en dehors du champ de vision. Ce système, basé uniquement sur l'utilisation d'une caméra et d'une paire de microphones, est fonctionnel à partir d'un apprentissage simple en ligne basé sur un petit nombre de pixels de peau détectés dans les premières trames de l'interaction.

Toutefois ce système pourrait gagner en précision : l'extraction de caractéristiques supplémentaires sur la vidéo pourrait permettre d'éviter un décrochement sur les détections en cas de mouvement d'un locuteur donné pendant qu'il continue de parler. L'extraction du mouvement entre 2 trames successives d'une détection pourrait permettre d'ajuster la prédiction. Qui plus est, l'ajout de modalités telles que la profondeur permettrait de se libérer de quelques suppositions : une segmentation de la profondeur pourrait permettre de garder une "zone d'interaction", où toutes les personnes se trouvant dans un certain rayon du robot est considéré comme participant à l'interaction en cours, tandis que les détections faites en dehors de ce rayon d'interaction sont rejetées.

Chapitre 4

Navigation dédiée à la détection d'humains à l'aide d'une fusion multimodale

Sommaire

4.1 Introduction 77
4.2 État de l'art sur la navigation en robotique 78
4.2.1 Les différents types de cartes 79
4.2.2 Notations 82
4.2.3 Cartographie pour la navigation 82
4.2.4 Localisation dans une carte 85
4.2.5 Planification de trajectoire 87
4.2.6 Synthèse de l'état de l'art 90
4.3 Une recherche d'humains reposant sur une fusion multimodale 91
4.3.1 Présentation de l'architecture 92
4.3.2 Description des types des données obtenues par les capteurs du robot . . 93
4.3.3 Construction des grilles "statiques" 96
4.3.4 Construction des grilles "dynamiques" 98
4.3.5 Construction de la grille de perception multimodale 101
4.3.6 Automate de décision 101
4.3.7 Planification de trajectoire 103
4.4 Résultats 106
4.4.1 Le robot Q.bo 106
4.4.2 Architecture logicielle 108
4.4.3 Conditions d'expérimentation 109
4.4.4 Réalisation des scénarios 115
4.5 Conclusion 124

4.1 Introduction

Dans ce chapitre, on propose d'utiliser l'information perçue par les capteurs pour guider la navigation d'un robot mobile. L'objectif est de permettre au robot de détecter les humains

de son environnement et de naviguer pour se positionner par rapport à eux (en restant à une certaine distance).

Il n'existe pas encore de robot capable de vivre en permanence avec les êtres humains, et de leur apporter une assistance dans leurs tâches quotidiennes de manière robuste. De nombreux facteurs entrent en jeu pour réaliser un tel système, tant en ce qui concerne le matériel (choix des capteurs et actionneurs) que les algorithmes mis en place (réalisation des tâches assignées au robot) ou les conventions sociales qu'un tel robot doit respecter pour être accepté par les personnes qui l'entourent. On se propose ici de réaliser et d'implémenter sur un robot réel un système de navigation autonome pour un robot mobile en environnement intérieur, dédié à la détection d'humains, reposant sur une fusion multimodale.

Le problème est donc le suivant : à partir de données capteurs de différentes natures, le robot doit construire et mettre à jour une carte de son environnement à l'aide de laquelle il prendra des décisions quant à ses propres déplacements. La carte doit contenir les éléments statiques de l'environnement du robot (mobilier, murs, etc..) mais aussi l'information sur la présence ou non d'humains. Le robot doit choisir de manière autonome son prochain objectif, planifier sa trajectoire, et l'exécuter en se localisant en permanence dans la carte de l'environnement.

Pour réaliser un tel système, on souhaite créer une instanciation de la boucle "perception-décision-action" présentée au chapitre 1, en proposant une représentation du monde adaptée à la recherche d'humains par un robot mobile via une fusion multimodale de cartes, et une stratégie de navigation simple permettant de naviguer vers un humain détecté.

Ce chapitre est ainsi divisé comme suit : un bref état de l'art sur la navigation est exposé dans la section 4.2, puis le système de navigation proposé est décrit dans la section 4.3. Enfin, la section 4.4 présente l'implémentation du système sur un robot réel et les résultats associés.

4.2 État de l'art sur la navigation en robotique

On souhaite permettre à un robot de se mouvoir de manière autonome au sein d'un environnement contenant des êtres humains. Pour cela, on s'appuiera sur des travaux existants en navigation robotique. La navigation autonome robotique consiste à doter un robot mobile équipé de capteurs de la capacité de se mouvoir sans intervention extérieure d'un point de l'espace à un autre sans collision. Il est donc nécessaire pour le robot d'être capable de construire une carte de son environnement, et de se localiser dans cette carte, comme cela est illustré par la figure 4.1.

La navigation autonome en robotique mobile se divise ainsi généralement en trois grands domaines conceptuels :

— La construction d'une représentation de l'environnement du robot, souvent appelée carte. La construction d'une telle représentation est appelée cartographie,
— La localisation, c'est à dire l'estimation de la position du robot au sein de cette carte,

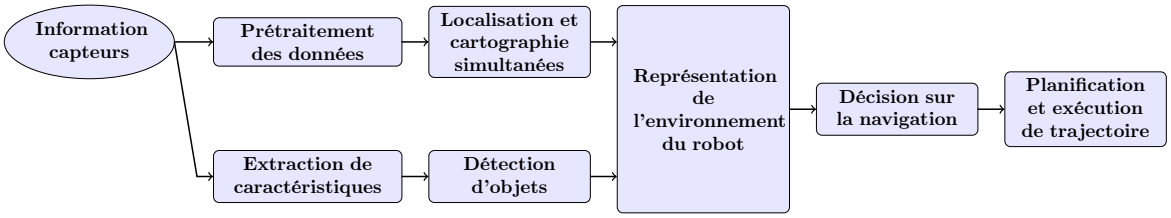


FIGURE 4.1 – Illustration des étapes d'un système de navigation autonome

— La planification et l'exécution de trajectoire, c'est à dire l'élaboration de la stratégie de déplacement sur la carte pour atteindre un objectif donné.

4.2.1 Les différents types de cartes

Pour se déplacer dans son environnement, un robot a recours à une représentation du monde qui l'entoure. C'est à cette carte qu'il se réfère pour prendre des décisions sur ses prochains déplacements, et pour se localiser. Elle contient généralement les éléments marquants de l'environnement du robot (par exemple les obstacles).

Il existe différents types de cartes dans la littérature [TBF05] : les cartes dites *topologiques* et les cartes *métriques*, elles-mêmes représentant plusieurs types de cartes différentes, comme le montre la figure 4.2.

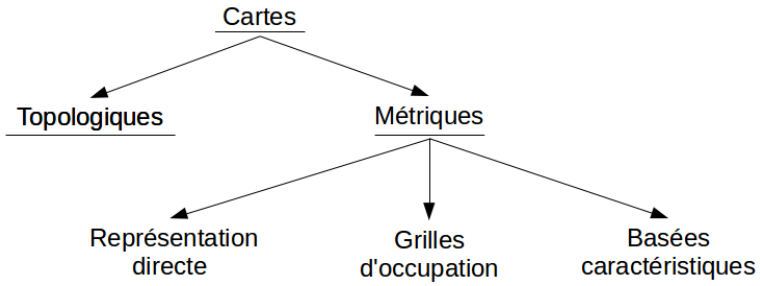


FIGURE 4.2 – Les différents types de cartes que peut construire un robot

4.2.1.1 Cartes topologiques

La **carte topologique**, introduite par [Kui78], représente l'environnement du robot sous forme d'un graphe dont les noeuds correspondent à des lieux "particuliers" de l'environnement du robot (par exemple, dans le cas d'un environnement intérieur, un tournant dans un couloir, une intersection en T, le centre d'un grand espace ouvert, etc...), et les arêtes connectent les noeuds entre lesquels une navigation directe est possible. Une telle carte est illustrée par la figure 4.3. Dans cet exemple, les noeuds sont les points de différentes couleurs reliés entre

eux par les arêtes bleues. L'environnement réel du robot est représenté en noir. L'avantage de ce type de carte réside dans la simplicité de sa représentation, qui est parcimonieuse : un environnement complexe peut être considéré sous une forme simple, sur laquelle il est facile de prendre une décision de navigation (le déplacement du robot revient directement à un déplacement dans un graphe, ce qui a été très exploré dans la littérature). Cependant une telle représentation perd la géométrie réelle de l'environnement du robot. De plus, les arêtes ne sont pas informatives sur le contenu du chemin, ce qui peut rendre un déplacement localement difficile.

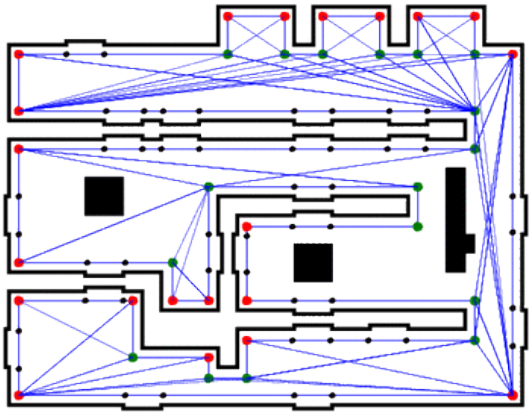


FIGURE 4.3 – Exemple d'une carte topologique [HB05]. Les noeuds sont les différents points rouges, verts et noirs, et les arêtes sont représentées en bleu. L'environnement réel du robot (plan d'architecte) est représenté en noir.

4.2.1.2 Carte Métrique

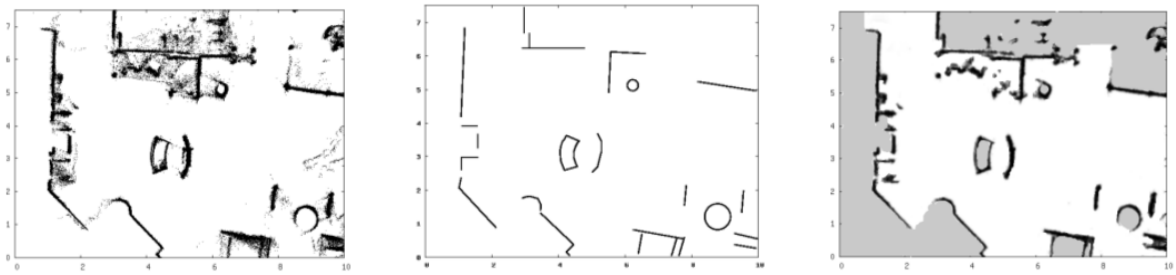
La **carte métrique**, comme son nom l'indique, représente les distances relatives des objets de la carte entre eux. On distingue trois grands types de cartes métriques : les cartes de représentation directe, les cartes "basées caractéristiques" (feature-based) et les grilles d'occupation. Un exemple de chacune de ces cartes est donné dans la figure 4.4

— Les **cartes de représentation directe** consistent à agglomérer directement les données brutes obtenues par les capteurs du robot. Elles sont parfois utilisées dans le cadre d'un robot équipé d'un capteur laser. Dans ce cas, la carte correspond à un nuage de points, chacun des points étant une des observations du robot. Ces cartes sont généralement peu précises car elles ne prennent pas en compte l'incertitude liée aux capteurs, mais ont l'avantage d'être rapides à construire.

— Les **cartes "basées caractéristiques"** extraient des caractéristiques géométriques des observations pour représenter les objets de l'environnement. Construire ces cartes correspond à estimer les paramètres des caractéristiques extraites. Dans ce cas, la carte est constituée de points de repère placés géométriquement de manière à représenter les distances réelles entre objets. Ce genre de carte a l'avantage de représenter la géométrie réelle de l'environnement du robot, mais ne modélise pas l'espace libre navigable. De

plus, seuls des points de repère paramétriques sont représentés (lignes, points, courbes, etc...).

— Les **grilles d'occupation**, introduites par [Elf89], consistent en une grille où chaque cellule correspond à une portion spatiale de l'environnement du robot. Chaque cellule contient une valeur comprise entre 0 et 1 indiquant son degré d'occupation (0 : la cellule est libre, 1 : la cellule est occupée). L'objectif est donc d'estimer à chaque instant d'observation l'état de chacune des cellules de la grille. Cette représentation a l'avantage de modéliser intégralement l'environnement du robot, et donc fait apparaître explicitement l'espace libre navigable, ce qui facilite la planification de trajectoire. Cependant une représentation dense comme celle-ci peut être coûteuse en ressources de calcul.



(a) Carte de représentation directe

(b) Carte basée caractéristiques

(c) Grille d'occupation

FIGURE 4.4 – Exemple des 3 types de cartes métriques sur le même jeu de données [Ayc10]

4.2.1.3 Synthèse sur les différents types de cartes

Pour choisir un type de carte, il faut s'intéresser au cadre d'utilisation et aux méthodes pour les construire. La littérature pour construire ces différents types de cartes est très vaste [TBF05 ; Dis+11], et les méthodes diffèrent grandement entre les cartes topologiques et les cartes métriques.

Pour construire les cartes topologiques, l'idée généralement retenue est de définir les points caractérisant un changement fort de l'environnement du robot. Ces points peuvent être extraits de manières diverses (exploitation d'un diagramme de Voronoï [Thr98], détection de coins de murs [HB05], création d'un plan de route probabiliste [Kav+96], etc..). Cependant, comme l'indique [Thr98], ces cartes ne sont pas suffisantes à elles seules pour permettre la navigation. Elles sont souvent associées à des représentations plus denses (cartes métriques) [TB96 ; SI09 ; SCI14] et utilisées pour des algorithmes de planification de trajectoire.

En ce qui concerne les cartes métriques, bien que souvent coûteuses en ressources de calcul, elles sont la représentation offrant le plus d'avantages pour la navigation robotique : représentation dense de l'espace, représentation de l'espace navigable, conservation de la géométrie

de la scène. Avec l'accroissement des capacités de calcul disponibles sur les robots, les grilles d'occupation sont devenues la représentation la plus populaire en robotique dans les travaux actuels. C'est ce genre de grilles que l'on utilisera pour représenter l'environnement du robot, et c'est sur les méthodes de construction de telles grilles que l'on va se concentrer dans la suite de ce chapitre.

4.2.2 Notations

Le processus de navigation d'un robot mobile est caractérisé par plusieurs grandeurs et peut être traité comme un filtre, prenant en entrée des observations fournies par des capteurs proprioceptifs et extéroceptifs et des commandes de déplacement données au robot, modifiant ainsi l'état du robot et celui de son environnement. Dans la suite, on utilisera donc les notations suivantes :

— z = $\{z_1, \cdots, z_t\}$: observations successives de l'environnement par le robot données par les capteurs extéroceptifs aux différents instants d'observation {1,$\cdots$,t}.
— u = $\{u_1, \cdots, u_t\}$: commandes successives données au robot, reçues directement sous forme numérique ou mesurées par des capteurs proprioceptifs (par exemple un odomètre)
— x = $\{x_1, \cdots, x_t\}$: états successifs (position) du robot dans son environnement.
— m = $\{m_1, \cdots, m_t\}$: carte des objets de l'environnement du robot aux différents instants d'observation. Si on utilise une carte déjà construite, contenant des objets statiques, on s'y référera simplement comme à la carte m.

Si toutes ces grandeurs sont parfaitement connues, ainsi que les erreurs qui leurs sont liées (incertitudes des mesures, qualité de la carte...), alors un robot mobile peut naviguer précisément dans un environnement et choisir ses déplacements sans erreur. Bien sûr, dans la majorité des cas, toutes ces grandeurs ne sont pas connues, et nécessitent d'être estimées. Ce sont les méthodes liées à l'estimation de telles grandeurs que l'on se propose de résumer maintenant.

4.2.3 Cartographie pour la navigation

4.2.3.1 Méthodes de construction de grilles d'occupation

Le processus de construction de grilles d'occupation, ou cartographie, a majoritairement été étudié du point de vue probabiliste. L'estimation directe de la carte suppose la connaissance à chaque instant de l'état du robot x_t, c'est à dire sa position dans la carte. Cette hypothèse n'est dans la majorité des cas pas respectée : elle nécessiterait d'informer "à la main" le robot de sa position à chaque observation. La construction de carte passe généralement par une estimation conjointe de la position du robot dans la carte et de la carte en elle-même, appelée SLAM (pour Simultaneous Localization And Mapping). Ainsi la construction d'une carte prend en entrée les observations capteurs et les commandes données au robot, et four-

nit en sortie une carte mise à jour et la position actualisée du robot dans cette carte (figure 4.5).

FIGURE 4.5 – Processus de cartographie

Le SLAM cherche donc à estimer la distribution de probabilités suivante à chaque observation :

$$p\left(m, x_t | z_{1:t}, u_{1:t}\right) \tag{4.1}$$

Le SLAM probabiliste est un des problèmes les plus étudiés de la robotique mobile. Le problème d'estimation d'état peut être résolu à l'aide d'un filtre bayésien générique. Cette estimation conjointe prend alors la forme suivante :

$$\underbrace{p(x_t, m | z_{1:t}, u_{1:t})}_{\substack{\text{estimée} \\ \text{à l'instant t}}} \propto \underbrace{p(z_t | x_t, m)}_{\substack{\text{correction} \\ \text{par modèle capteur}}} \cdot \underbrace{\int_{x_{t-1}} \underbrace{p(x_t | x_{t-1}, u_t)}_{\text{modèle dynamique}} \cdot \underbrace{p\left(x_{t-1}, m | z_{1:t-1}, u_{1:t-1}\right)}_{\substack{\text{estimée} \\ \text{à l'instant t-1}}}}_{\text{prédiction}} \tag{4.2}$$

Cette forme est très proche du filtre bayésien vu dans la section 3.2, avec une prédiction et une correction à l'aide de modèles capteurs. Comme on peut le voir, cette forme nécessite de connaître deux distributions :

— $p(x_t|x_{t-1},u_t)$, soit la distribution de probabilité sur l'état x_t du robot à l'instant t, connaissant son état estimé x_{t-1} à l'instant t-1 et la commande de déplacement u_t à l'instant t. C'est le modèle de mouvement du robot, ou **modèle dynamique**. Il est souvent généré à partir de données odomètriques. Une hypothèse de Markov est généralement acceptée, ce qui signifie qu'on considère que l'état à l'instant t ne dépend que de l'état à l'instant t-1 et de la commande envoyée, et non des instants précédents. La différence avec le modèle dynamique présenté dans la section 3.2 est l'ajout du déplacement du robot u_t. En effet, la nouvelle position x_t à l'instant t dépend de la commande envoyée au robot pour son déplacement.

— $p(z_t|x_t,m)$, soit la distribution de probabilités sur les observations z_t à l'instant t, connaissant la position du robot x_t à l'instant t et la carte m. C'est **le modèle capteur** du robot. Il nécessite d'avoir caractérisé les capteurs du robot pour prendre en compte les incertitudes qui leur sont liées.

Pour modéliser au mieux des distributions non paramétriques, comme c'est souvent le cas pour les modèles capteur et de mouvement, on a vu apparaître des travaux utilisant les filtres particulaires [Aru+02], avec les SLAM à "maximum de vraisemblance" [VAA07 ; MT07].

L'idée ici est de construire une carte de manière incrémentale, en ajoutant les observations à la carte au fur et à mesure. La méthode consiste à rechercher la position $\widehat{x}_t$ qui maximise, sur l'ensemble des états possibles :

$$\widehat{x}_t = \underset{x_t}{\operatorname{argmax}}\ p\left(x_t | z_t, u_t, \widehat{x}_{t-1}, \widehat{m}_{t-1}\right) = \underset{x_t}{\operatorname{argmax}}\ \{p(z_t | x_t, \widehat{m}_{t-1}) p(x_t | u_t, \widehat{x}_{t-1})\} \tag{4.3}$$

avec $p(z_t|x_t,\widehat{m}_{t-1})$ la probabilité de l'observation la plus récente connaissant la carte estimée précédemment et la position supposée du robot, $p(x_t|u_t,x_{t-1})$ étant le modèle dynamique. Cette maximisation cherche à répondre à la question "connaissant les estimées de cartes et de position à l'instant précédent, ainsi que les observations et la commande donnée à l'instant t, quelle est la position la plus vraisemblable à l'instant t ?". Calculer $p(z_t|x_t,\widehat{m}_{t-1})$ représente une opération dite de "scan-matching". De nombreux algorithmes ont vu le jour pour procéder au scan-matching sur les différents types de cartes [LM97 ; HS10].

Une fois la position la plus vraisemblable estimée, la carte est mise à jour en y ajoutant le scan directement :

$$m_t = \widehat{m}_{t-1} \cup \{< \widehat{x}_t, z_t >\} \tag{4.4}$$

Pour pouvoir ajouter les observations à l'instant t, il est nécessaire de construire une carte à partir de données capteurs. C'est la distribution de probabilités $p(m|x_t,z_t)$, aussi appelée **modèle capteur inverse**. Elle dépend des caractéristiques des capteurs utilisés et de la manière dont ils perçoivent l'environnement du robot.

Cette approche de type "en ligne" est très performante lorsqu'il s'agit de maintenir une carte en temps réel (type carte avec oubli). Cependant, comme à chaque instant seule la position la plus vraisemblable est considérée, et non la trajectoire du robot, un endroit déjà observé ne pourra pas être reconnu comme tel.

Pour remédier à ce problème d'autres filtres ont vu le jour, notamment des filtres particulaires Rao-Blackwellisés [Mur99], permettant la factorisation suivante :

$$\underbrace{p(x_{1:t}, m | z_{1:t}, u_{1:t})}_{\substack{\text{Estimation conjointe à l'instant t} \\ \text{de la carte et de la trajectoire}}} = \underbrace{p(m | x_{1:t}, z_{1:t})}_{\substack{\text{Estimation de la carte} \\ \text{pour une trajectoire donnée}}} \cdot \underbrace{p(x_{1:t} | z_{1:t}, u_{1:t-1})}_{\substack{\text{Proposition de} \\ \text{trajectoires possibles}}} \tag{4.5}$$

Dans ce cas, ce sont des trajectoires entières qui sont estimées : on a donc une carte estimée pour chacune des trajectoires possibles, ce qui permet de reconnaître des endroits déjà visités. Une telle approche a été adoptée aussi bien dans le cas de cartes basées caractéristiques [Mon+02 ; Thr+04] que dans le cas de grilles d'occupation [Hah+03 ; GSB07]. Cette résolution est appelée problème de SLAM "complet".

4.2.3.2 Types de capteurs employés

Des algorithmes de SLAM ont été conçus pour tous types de capteurs, l'unique contrainte étant bien sûr d'obtenir une distance des objets de l'environnement au robot. Le capteur le plus utilisé en robotique mobile reste le télémètre laser, et la plus grande partie des travaux effectués jusqu'à présent considère principalement ce genre de données, mais on voit apparaître d'autres types de capteurs pour le SLAM, qui augmentent le nombre de possibilités, tels que les capteurs RGB-D, permettant de faire du SLAM 3D [New+11 ; Whe+13], ou la stéréovision [JL05]. D'autres applications ont ajouté des contraintes, comme le SLAM monoculaire [GD15 ; LLS05].

D'autres types de représentations ont été utilisés pour construire des cartes, ajoutant des données de plus haut niveau que la simple occupation [MCB11 ; Rei11], où le cadre évidentiel est employé pour produire des cartes contenant de l'information quant au contenu sémantique de la cellule. Dans le cas de [Kur+12], des données extraites de cartes externes sont utilisées pour extraire de l'information sur le type d'objets entourant le robot (immeuble cartographié, infrastructure inconnue, etc...).

4.2.3.3 Synthèse sur la cartographie

La construction de grilles d'occupation a fait l'objet de nombreux travaux dans la littérature, avec deux approches principales : le SLAM "en ligne" consistant en une agglomération des informations observées et le SLAM "complet" qui construit une carte consistante d'un endroit. Dans le présent travail, il est possible d'utiliser les deux types d'approches : en effet, un robot évoluant avec des êtres humains aura un périmètre d'action délimité dans un endroit *a priori* connu pour ce qui est des objets statiques (murs, meubles, etc...). Ce genre de situation est adapté à une cartographie "complète" et permettra au robot de se localiser facilement dans son environnement. Toutefois, pour qu'il remplisse sa mission de détection d'humains, il est nécessaire d'augmenter le pouvoir de représentation des grilles via la fusion capteur. Pour cela, une approche "en ligne" semble plus adaptée, car elle permet d'ajouter de l'information temporaire (avec oubli) dans la carte.

4.2.4 Localisation dans une carte

Le problème de la localisation dans une carte est un sous-problème du SLAM et a lui aussi été largement exploré dans la littérature. Le principe est le suivant : connaissant une carte m des objets statiques de l'environnement du robot, on cherche à estimer à chaque instant la position x_t du robot dans son environnement, comme le montre la figure 4.6.

De nouveau, la localisation a été très étudiée du point de vue probabiliste. Cela revient à estimer à chaque instant la distribution suivante :

FIGURE 4.6 – Processus de localisation au sein d'une carte

$$p(x_t|u_t, z_t, m) \tag{4.6}$$

Ce problème est plus simple que celui du SLAM, la carte m étant connue (on ne cherche donc pas à l'estimer). [TBF05] présente pour résoudre ce problème une réalisation directe du filtre bayésien. Elle met à jour la position estimée du robot à chaque instant en utilisant l'estimée à l'instant précédent, comme le montre l'équation 4.7. Une telle utilisation directe du filtre bayésien est appelée localisation de Markov.

$$\underbrace{p(x_t|u_t, z_t, m)}_{\substack{\text{distribution } a \text{ } posteriori \\ \text{à l'instant t-1}}} \propto \underbrace{p(z_t|x_t, m)}_{\substack{\text{mise à jour} \\ \text{par modèle capteur}}} \cdot \underbrace{\int_{x_{t-1}} \underbrace{p(x_t|u_t, x_{t-1}, m)}_{\text{modèle dynamique}} \underbrace{p(x_{t-1}|u_{t-1}, z_{t-1}, m)}_{\substack{\text{distribution } a \text{ } posteriori \\ \text{à l'instant t-1}}}}_{\text{prédiction}} \tag{4.7}$$

Pour réaliser un tel filtre, la connaissance des deux modèles classiques est nécessaire : le modèle capteur et le modèle dynamique du robot. Différentes implémentations de ce filtre existent dans la littérature : [TBF05 ; CHMM12] proposent par exemple une utilisation du filtre de Kalman étendu (EKF) [AM79], où des distributions gaussiennes sont utilisées pour les modèles, le bruit, et pour la distribution *a posteriori* à calculer. La distribution *a posteriori* est mise à jour à chaque itération en calculant ses deux paramètres : sa moyenne et sa variance. Cette technique est très utilisée dans le cadre de cartes "basées caractéristiques".

Toutefois, une des implémentations les plus populaires prend la forme d'un filtre particulaire [Aru+02]. Cette implémentation, intitulée localisation de Monte-Carlo, et introduite par [Fox+99], maintient à chaque instant d'observation un ensemble $x_t = \{x_t^{[1]}, \cdots, x_t^{[M]}\}$ de particules, correspondant à l'ensemble des positions probables du robot dans la carte. Cette implémentation présente de nombreux avantages : l'utilisation d'un filtre non-paramétrique permet de modéliser les capteurs et la dynamique du robot avec beaucoup de liberté, et le temps de calcul d'un tel filtre est très rapide. Cette approche est adaptable à différents types de données. Elle est particulièrement performante dans le cas d'utilisation d'un LIDAR [Fox+99], mais a aussi été utilisée dans le cadre d'autres capteurs : caméra [Thr+01], réseau de capteurs sans fil [BL08], Kinect [FJL12] etc...

La plus grande difficulté tient dans les conditions d'initialisation du filtre. En effet, pour

initialiser la localisation de Monte-Carlo il est nécessaire de connaître la distribution *a priori* $p(x_1)$, nécessaire à l'estimation. On peut distinguer trois cas :

- Suivi de position : dans ce cas, on suppose que la position initiale du robot est connue. On peut donc initialiser le filtre avec une distribution centrée sur la position initiale (la forme du filtre étant dépendante de la précision de notre connaissance de la position initiale du robot).
- Localisation globale : dans ce cas, la position du robot est inconnue. On initialise donc le filtre avec une distribution uniforme sur l'ensemble des cellules. Lors de la première itération, ce sont les observations qui donneront la distribution de probabilités *a posteriori* sur l'état du robot
- Autre initialisation : quelle que soit la connaissance de la position initiale du robot, elle peut se transcrire directement géographiquement dans une distribution sur une grille d'occupation en attribuant une distribution uniforme aux cellules de position initiale possibles et nulle ailleurs.

Plus récemment, d'autres méthodes de localisation ont fait leur apparition. On peut notamment citer [SI09 ; SCI14], qui utilisent des méthodes issues de la littérature sur la saillance visuelle (extraction des éléments les plus pertinents de la scène via des caractéristiques de vision par ordinateur) pour se localiser dans une carte topologique, à l'aide de "landmark-matching", c'est à dire en comparant les caractéristiques extraites à celles stockées dans la carte topologique.

Toutefois ces méthodes sont généralement plus coûteuses en ressources de calcul, et ne sont adaptées qu'à des observations vidéo (il est par exemple difficile de définir une notion de saillance dans un scan laser). La localisation de Monte-Carlo étant rapide, et idéale pour des grilles d'occupation, c'est celle qu'on utilisera dans la suite pour se localiser au sein d'une carte.

4.2.5 Planification de trajectoire

La planification de trajectoire robotique est le processus de calcul des positions successives à prendre pour atteindre un but dans la carte m, à partir de la position actuelle du robot. On considère ici que le robot se déplace dans une grille d'occupation. Si le robot dispose d'une carte des objets statiques de son environnement, il peut désirer se déplacer vers un but hors de son champ de perception actuel. Dans ce cas, la planification est souvent divisée en deux sous-parties :

- **La planification globale**, qui consiste à choisir son chemin directement sur la carte, sans se soucier de possibles obstacles réels non signalés sur celle-ci. C'est l'équivalent d'une personne choisissant son chemin sur la carte d'une ville.
- **L'évitement d'obstacles et l'exécution du parcours**, qui correspond à la gestion d'événements imprévus lors de la réalisation du chemin global calculé dans l'étape précédente. C'est l'équivalent d'une personne contournant les piétons se trouvant sur son chemin dans la rue.

Des algorithmes différents existent pour les deux types de planification.

4.2.5.1 Planification globale

La planification globale consiste à trouver un chemin adapté pour le robot dans l'espace libre navigable pour aller d'une cellule m^i (position actuelle du robot dans la carte), à une cellule m^j (but à atteindre). Une grande majorité des algorithmes de calcul utilise des méthodes issues de la théorie des graphes. Les deux algorithmes de planification les plus utilisés sont les algorithmes de Dijkstra [Dij59] et A* [HNR68], et sont des algorithmes de plus court chemin au sein d'un graphe. Ils reposent sur une heuristique simple et restent les plus efficaces dans le cas de graphes de petites tailles, bien qu'ils puissent amener dans le pire des cas à une exploration exhaustive du graphe. D'autres algorithmes de plus court chemin ont été utilisés pour la planification de trajectoire tels que l'algorithme de colonies de fourmis [LHL09], ou l'algorithme génétique [Cha+12]. La théorie des graphes s'applique quasiment directement au calcul de trajectoire.

Pour ce qui est de la représentation de la carte en matière de graphe, il existe deux types de planificateur de trajectoire globale :

- Les planificateurs considérant l'intégralité de la carte comme un graphe. Dans ce cas, chaque cellule est considérée comme étant un noeud du graphe, et des arêtes sont créées entre chaque cellule et ses plus proches voisins, avec un poids de 1. Le problème d'une telle représentation peut être la taille, dans le cas d'une carte de très grande ampleur.
- Les planificateurs diminuant la taille du problème : dans ce genre de situation, les algorithmes évoqués dans la section 4.2.3 deviennent intéressants. Pour transformer une grille d'occupation en représentation de taille moindre, plusieurs méthodes sont disponibles : la méthode "probabilistic roadmap" (PRM) [Kav+96] sélectionne des points au hasard dans l'espace libre navigable (avec une probabilité plus forte dans les espaces encore non explorés) pour construire le graphe et s'arrête lorsque le graphe contient des noeuds de manière assez dense dans l'ensemble de la carte. Un exemple est donné par la figure 4.7. Une autre méthode consiste à obtenir un squelette de la carte d'occupation, soit l'ensemble des points de la grille d'occupation se trouvant à la plus grande distance possible des objets qui la composent. Le diagramme de Voronoï généralisé est souvent utilisé. Un exemple de squelette est illustré par la figure 4.8.

Les algorithmes présentés ici ne prennent en compte dans leur forme originale que le plus court chemin. Cependant, en robotique, le plus court chemin n'est pas toujours le chemin souhaité. En effet, un robot qui longe les murs pour gagner du temps ou qui coupe la route d'un être humain n'a pas un comportement adapté. Ces considérations n'ont commencé à être prises en compte que récemment [Spa15].

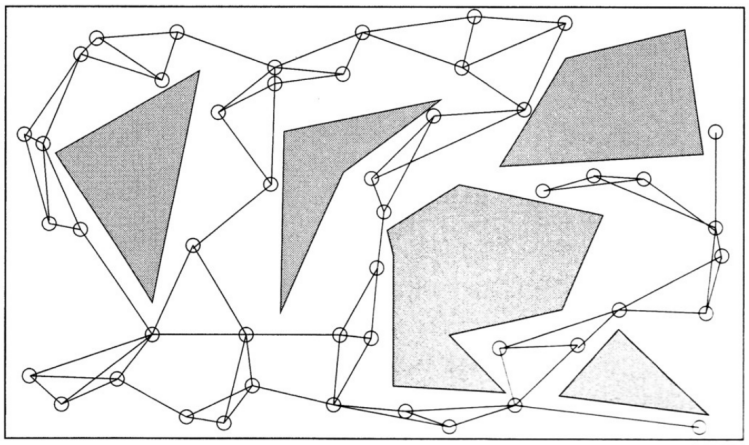


FIGURE 4.7 – Exemple d'un graphe construit à l'aide de la méthode PRM de [Kav+96]

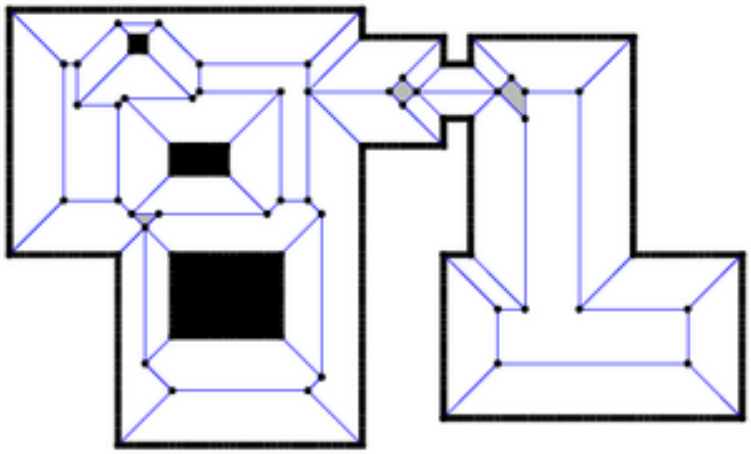


FIGURE 4.8 – Exemple d'un graphe construit à l'aide d'un diagramme de Voronoï [HB05]

4.2.5.2 Évitement d'obstacles et exécution de parcours

La planification globale étant effectuée sur une représentation statique du monde, le chemin décidé ne prend en compte ni les obstacles dynamiques pouvant apparaître sur la trajectoire du robot, ni l'apparition de nouveaux objets statiques qui n'appartiennent pas à la carte de référence.

Pour éviter de tels obstacles, une stratégie de navigation locale prenant en compte les observations effectuées par le robot est nécessaire. Des algorithmes de différentes sortes existent pour résoudre ce problème. Une des solutions les plus populaires est l'utilisation des champs de potentiel : dans ce cas, les cases de la grille occupées par un obstacle sur le chemin du robot reçoivent un fort potentiel, tandis que le but à atteindre reçoit un potentiel faible, créant ainsi

une carte constituée de "bosses" et de "creux", les creux étant plus attractifs pour le robot, et les bosses étant infranchissables, comme cela est montré par la figure 4.9. En suivant le gradient du potentiel dans ce genre de représentation, le robot pourra ainsi contourner les obstacles. Une méthode de ce type a notamment été utilisée dans [Kha85 ; KS06]. Les champs de potentiel sont une solution élégante à l'évitement d'obstacle, mais peuvent avoir un haut coût de calcul (nécessité de calculer le gradient du potentiel).

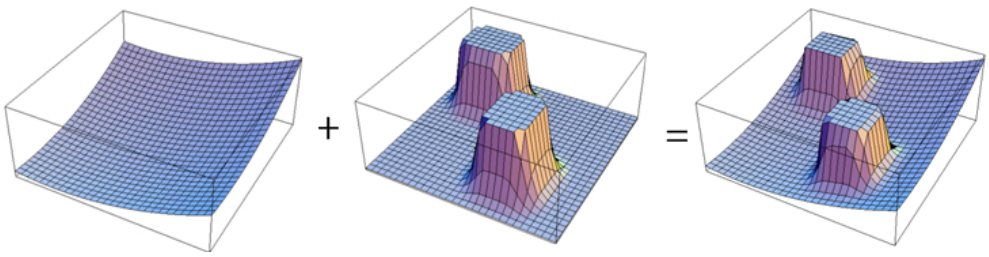


FIGURE 4.9 – Exemple d'un champ de potentiel destiné à l'évitement d'obstacles. La grille de gauche représente l'attractivité d'une partie de la carte (les points ayant le potentiel le plus bas). Les obstacles sont représentés dans la grille du milieu par les "montagnes". L'objectif du robot se trouve en bas de la pente, et le robot lui-même se trouve en haut.

D'autres méthodes ont été mises en place, telles que la DWA (Dynamic Window Approach) [FBT97], qui projette des trajectoires et les évalue selon des critères de proximité aux obstacles, à l'objectif, et en fonction de la vitesse du robot. Les trajectoires impliquant une collision sont rejetées. Cette approche, dont l'implémentation est librement disponible, est l'une des plus utilisées.

Une revue plus détaillée des différents types d'algorithmes pour l'évitement d'obstacles peut être trouvée dans [Kun+06].

4.2.6 Synthèse de l'état de l'art

Il existe de nombreuses méthodes pour la réalisation d'un système de navigation autonome, avec l'utilisation conjointe de la cartographie ; de la localisation et de la planification de trajectoire. Cependant, il existe encore peu de systèmes proposant une cartographie reposant sur la fusion de données provenant de sources différentes (laser, caméra, microphones).

On propose dans le présent chapitre d'utiliser des méthodes existantes pour intégrer directement l'information de présence d'êtres humains dans des grilles, pour permettre au robot de prendre des décisions sur des grilles au pouvoir de représentation supérieur à une grille d'occupation classique, via une stratégie simple de gestion d'information prioritaire.

4.3 Une recherche d'humains reposant sur une fusion multimodale

Comme il a été évoqué dans la section 4.1, on souhaite permettre à un robot de naviguer de manière autonome dans un environnement intérieur (appartement, maison...) contenant des humains. Il doit pouvoir détecter et localiser les humains qui l'entourent dans cet environnement, et se positionner par rapport à eux. Un scénario possible ici peut être le suivant : le robot entre dans une pièce, explore les différentes parties de cette pièce et perçoit un humain en cours d'exploration. Il se déplace alors pour se positionner à un mètre de l'humain, prêt pour une interaction ultérieure. Pour réaliser un tel système, il est nécessaire de faire appel à l'intégralité de la chaîne "perception-décision-action".

Dans la section 4.2, il a été montré que l'élément central pour un système de navigation autonome est la représentation de l'environnement, soit la carte utilisée par le robot pour prendre des décisions quant à ses prochains déplacements. L'intérêt des grilles métriques pour une telle représentation a été souligné : elles sont une représentation dense, contenant l'information d'espace libre navigable.

Toutefois, on souhaite ici inclure directement dans la carte d'autres informations que le robot doit prendre en compte lors de son exploration de la scène. La représentation de son environnement doit donc contenir deux types d'informations :

- **Les informations statiques**, liées aux caractéristiques générales de son espace de travail : murs, gros mobilier, objets immobiles de l'environnement, etc... Ces objets correspondent à l'environnement connu *a priori* par le robot
- **Les informations dynamiques**, liées aux objets mobiles de la scène : humains, chaises, cartons, tous types d'obstacles qui ne sont pas à la même place à chaque passage du robot.

Le robot a dans un premier temps été équipé de deux types de capteurs : un capteur RGB-D, produisant des données vidéo et de profondeur, ainsi qu'une paire de microphones donnant accès au flux audio. Cependant, les données de profondeur produites par le capteur RGB-D ne sont pas assez précises pour produire des grilles consistantes dans le temps. Ainsi, pour améliorer la robustesse du système, un télémètre laser a plus tard été ajouté. Ceci permet d'avoir une information de profondeur plus précise.

Pour réaliser la carte souhaitée, on propose de créer plusieurs grilles métriques contenant les différentes informations extraites des capteurs, et des les fusionner de manière simple, pour obtenir une représentation complète de l'environnement du robot. Ceci permettra au robot de prendre une décision directement sur la carte, dépendant des informations capteurs obtenues. Pour la prise de décision, un automate accordant à chaque modalité un niveau de priorité adéquat est utilisé. Finalement la planification de trajectoire est effectuée à l'aide d'un diagramme de Voronoï.

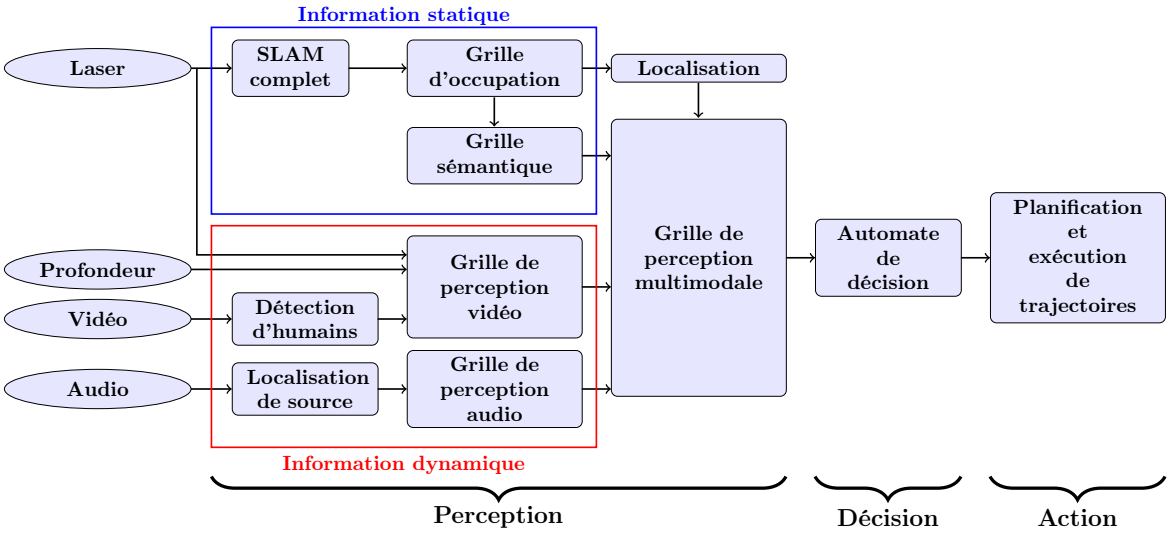


FIGURE 4.10 – Architecture proposée pour la navigation multimodale : l'information statique est extraite "hors ligne", lors d'un passage du robot dans l'environnement, à l'aide du SLAM complet. L'information dynamique est l'information apportée "en ligne" lorsque le robot se déplace.

4.3.1 Présentation de l'architecture

Le système de navigation proposé est présenté dans la figure 4.10. La perception est divisée en deux parties : l'extraction d'informations statiques, et l'extraction d'informations dynamiques.

Les informations statiques sont extraites au cours d'un premier passage du robot, lors d'une cartographie de l'environnement à l'aide d'une méthode de SLAM complet. Ceci permet d'avoir accès à une grille d'occupation qui permettra au robot de se localiser au cours du temps. Cette grille sera aussi utilisée pour la création de la grille dite "sémantique", qui contient les points d'intérêt "par défaut" pour le robot.

Les informations dynamiques sont extraites des flux laser, audio, vidéo et de profondeur au cours de l'exploration du robot. Deux grilles métriques de perception sont créées, dans une méthode similaire au SLAM à maximum de vraisemblance. L'une regroupe les informations du capteur RGB-D et du capteur laser, et l'autre les informations du capteur audio.

Finalement ces deux grilles sont fusionnées au sein d'une grille de perception multimodale, qui correspond à la représentation finale de l'environnement du robot. La décision quant à la prochaine destination du robot est prise à l'aide d'un automate binaire, et la planification de trajectoire a lieu aux deux niveaux présentés dans la section 4.2 : tout d'abord une trajectoire globale est élaborée à l'aide de la position actuelle du robot et de la grille d'occupation, puis la trajectoire est exécutée avec une méthode d'évitement d'obstacles pouvant apparaître sur le chemin du robot.

On considère ici qu'à chaque temps d'observation, une détection d'êtres humains est faite sur la vidéo, et une localisation de source sonore (telle qu'on a pu la décrire dans le chapitre 3) est effectuée sur le flux audio.

4.3.2 Description des types des données obtenues par les capteurs du robot

4.3.2.1 Capteur RGB-D

Il existe plusieurs capteurs de types RGB-D, qui renvoient tous le même type de données. Un capteur RGB-D (figure 4.11) donne accès à deux flux différents : le flux de données de profondeur et le flux vidéo RGB.

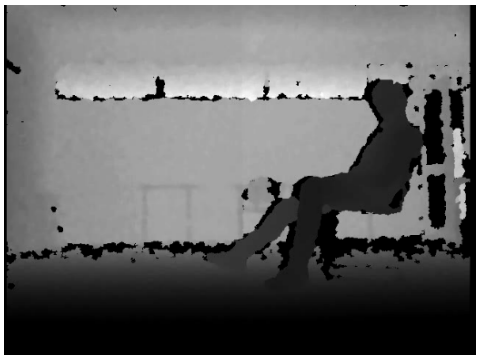
FIGURE 4.11 – Capteur RGB-D Asus Xtion Pro live

Le flux RGB produit une vidéo couleur classique (figure 4.12 (a)), soit un ensemble de pixels correspondant chacun à une partie de la scène filmée par le capteur. L'image de profondeur produit une image de mêmes dimensions que l'image couleur, correspondant à la même scène, qui pour chaque pixel contient une valeur correspondante à la distance de l'objet contenu par ce pixel à la caméra. Pour la visualisation, cette image est souvent retranscrite en niveaux de gris (figure 4.12 (b)) : plus l'objet face à la caméra est proche, plus il est sombre à l'image. Inversement, plus l'objet est loin, plus il est clair. Ce genre de capteur étant limité en distance de perception proche et lointaine, les pixels se trouvant hors du champ de perception représentés en noir.

Les capteurs RGB-D sont équipés de différentes technologies pour pouvoir produire l'image de profondeur (temps de vol, lumière structurée [Sha+14]), mais les données prennent toujours la même forme. Ainsi il est possible de placer un objet détecté dans l'image de profondeur dans le repère de la caméra (figure 4.13).

Pour cela il est nécessaire de connaître la focale horizontale (resp. verticale) f_x (resp. f_y) de la caméra, ainsi que son centre optique horizontal (resp. vertical) c_x (resp. c_y). Ces valeurs peuvent être obtenues lors d'une calibration de l'appareil RGB-D. Une fois ces valeurs connues, on peut transformer un pixel de coordonnées [u,v], donné à une distance d dans l'image de profondeur, en un point de coordonnées x, y, z dans le repère de la caméra, grâce aux équations suivantes :

(a) Image RGB tirée d'un capteur Asus Xtion Pro live



(b) Image de profondeur tirée d'un capteur Asus Xtion Pro live

FIGURE 4.12 – Exemple de données extraites d'un capteur Asus Xtion Pro live

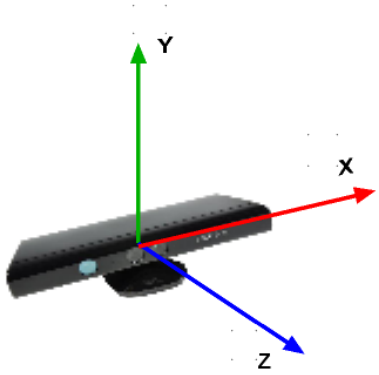


FIGURE 4.13 – Système de coordonnées du capteur RGB-D (ici type Kinect)

$$z = d \tag{4.8}$$

$$x = \frac{(u - c_x) \cdot z}{f_x} \tag{4.9}$$

$$y = \frac{-(v - c_y) \cdot z}{f_y} \tag{4.10}$$

Ce genre de données a ses limites : les images de profondeur fournies par les capteurs RGB-D ne sont pas parfaites, et on voit généralement apparaître des phénomènes d'ombres (contours de l'être humain sur la figure 4.12 (b)). Ces capteurs ont des portées plutôt courtes (entre 5 et 10 mètres en général), et une portée minimale (impossible de percevoir la distance en dessous d'une distance donnée, généralement de l'ordre d'une cinquantaine de centimètres).

4.3.2.2 Capteur laser

Les capteurs laser (figure 4.14), ou télémètres laser, sont des capteurs permettant de mesurer les distances entre le robot et les obstacles de l'environnement.

FIGURE 4.14 – Capteur laser Hokuyo URG-04-LX

Pour obtenir des distances, le capteur balaie son environnement à l'aide d'un rayon laser, émis à une certaine longueur d'onde dans un plan. Le rayon se propage en ligne droite, et lorsqu'il atteint un obstacle, est réfléchi vers le capteur. Le capteur calcule alors la différence de phases entre le rayon émis et le rayon reçu et en déduit la distance de l'obstacle. Un nombre fini de rayons est émis à chaque instant d'observation, dépendant de la résolution angulaire du capteur, ainsi que de son champ de perception. L'ensemble de ces rayons est appelé scan laser, et donne accès à un nombre de distances inférieur ou égal au nombre de rayons émis (dans certains cas, aucun obstacle n'est rencontré par le rayon). Ainsi, pour un rayon donné, d'angle d'incidence θ et de distance de point d'impact d, on peut calculer les coordonnées (x,y,z) de son point d'impact dans le repère du capteur laser (figure 4.15) :

$$x = d \cdot cos(\theta) \tag{4.11}$$
$$y = d \cdot sin(\theta) \tag{4.12}$$
$$z = 0 \tag{4.13}$$

Le capteur balayant son environnement dans un plan, la cordonnée z est toujours égale à 0.

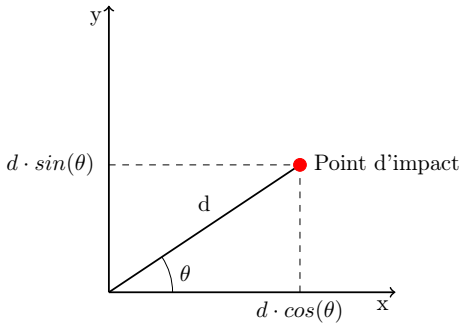


FIGURE 4.15 – Calcul des coordonnées d'un point d'impact du scan laser

4.3.3 Construction des grilles "statiques"

Comme on l'a dit, les grilles statiques, soit la grille d'occupation et la grille sémantique, contiennent des informations sur les objets immobiles de l'espace de travail du robot. Elles sont toutes deux construites en amont de l'exploration du robot, et stockées en mémoire. Pour illustrer la construction des différentes grilles, on propose d'utiliser un exemple jouet représentant un faux espace de travail du robot, présenté à la figure 4.16.

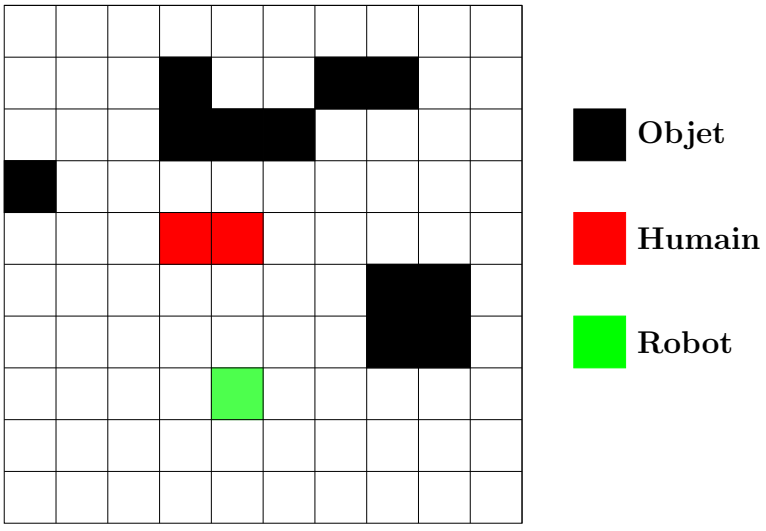


FIGURE 4.16 – Exemple jouet d'une scène où peut se retrouver le robot

4.3.3.1 Grille d'occupation

La grille d'occupation est une grille classique produite grâce à une méthode de SLAM complet décrite dans la section 4.2.3. Elle est construite à l'aide d'une navigation téléguidée du robot dans l'espace de travail sans humains, et contient les objets statiques principaux qui le composent. La construction d'une telle grille n'est pas une contrainte forte dans le cas d'un

robot d'intérieur amené à toujours travailler dans le même espace de travail. Elle nécessite une exploration unique avant la mise en autonomie du robot, qui permet à celui-ci d'acquérir une sorte de "plan d'architecte" de son environnement.

Cette grille sera utilisée à deux fins principales :

— **La localisation du robot :** comme il a été décrit dans la section 4.2.4, il est possible de localiser un robot avec une grille d'objets statiques et des observations de capteurs. Connaître sa position à chaque instant dans un repère absolu est nécessaire pour remplir une mission d'exploration d'un environnement.

— **La planification de trajectoire :** dans la section 4.2.5, on a vu que la planification de trajectoire vers un objectif donné se fait généralement en deux étapes : la planification globale, puis l'exécution du trajet et l'évitement d'obstacles. La grille d'occupation sera donc utilisée pour la planification globale.

Pour construire cette carte, l'algorithme proposé par [GSB07], basé sur un filtre particulaire Rao-Blackwellisé a été utilisé. Il repose sur l'utilisation d'un capteur laser, et construit une carte consistante de l'environnement statique du robot. La grille d'occupation correspondant à l'exemple jouet est illustrée par la figure 4.17. Elle contient une information seuillée binaire : chaque cellule reçoit une valeur 1 dans le cas où elle contient un objet de l'environnement et 0 dans le cas où elle contient de l'espace libre navigable par le robot.

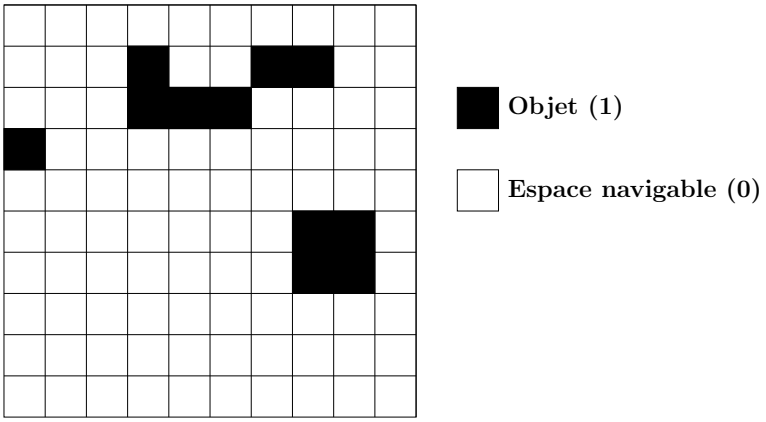


FIGURE 4.17 – Grille d'occupation de l'exemple jouet

4.3.3.2 Grille sémantique

Lorsque le robot entre dans son espace de travail, il n'est pas certain qu'il soit en mesure de percevoir directement des humains. Il devra donc commencer à explorer la scène sans être guidé par une détection particulière. On souhaite que le robot n'ait pas à explorer la scène de manière aléatoire. En effet, il y a des endroits dans un environnement qu'il est plus naturel pour le robot d'aller explorer en priorité que d'autres : ce sont les endroits où il y a le plus de chances de trouver des êtres humains (devant le poste de télévision, autour de la table du dîner, près du lit, etc....). On souhaite donc créer une grille contenant les endroits qu'il est

intéressant d'aller visiter "par défaut". Elle porte le nom de grille sémantique, car elle contient des zones intéressantes du fait de leur contenu.

Il est possible de déterminer ces endroits de deux manières différentes : les définir "à la main", en désignant les zones intéressantes, puis les stocker directement dans la mémoire du robot, ou les apprendre au fur et à mesure du passage du robot dans l'espace de travail (à chaque fois qu'un humain est détecté, on garde en mémoire sa localisation dans la carte). Les zones de la carte contenant une grande densité d'humains détectés sont les zones qui apparaîtront par la suite dans la grille sémantique. Dans ce chapitre, on choisit de stocker les zones intéressantes "à la main" dans la mémoire du robot.

La grille sémantique construite sur l'exemple jouet est visible sur la figure 4.18. Elle correspond à une grille binaire : les cellules se trouvant dans une zone d'intérêt sont mises à 1 et les autres sont mises à 0. On peut par exemple imaginer que le carré "d'objet" de la grille d'occupation représente une table autour de laquelle il y a de fortes chances de retrouver des êtres humains, et que la forme en haut de la grille représente un canapé. Ces zones sont naturellement intéressantes car on peut y trouver des personnes. Les zones d'intérêt peuvent aussi bien se trouver à coté d'obstacles de l'environnement statiques qu'au même endroit.

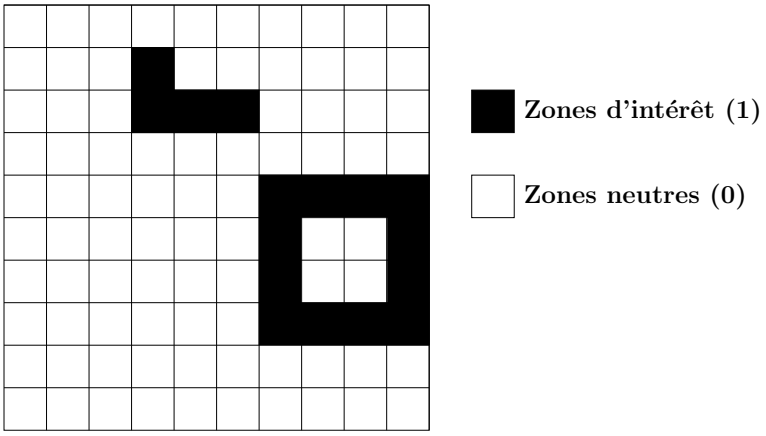


FIGURE 4.18 – Grille sémantique pour l'exemple jouet

4.3.4 Construction des grilles "dynamiques"

Les grilles dynamiques contiennent l'information perçue lors du passage du robot pour la recherche d'humains. A chaque instant d'observation, on dispose d'un scan laser, de l'image de profondeur, de l'image vidéo couleur, et du flux audio. On propose de créer deux grilles à chaque instant d'observation : la grille de perception vidéo et la grille de perception audio.

Comme on estime la position x_t du robot à chaque instant t d'observation, on choisit une approche similaire au SLAM à maximum de vraisemblance : connaissant la position la plus vraisemblable du robot dans la carte, on construit la carte à partir du capteur. Pour cela il est nécessaire d'utiliser un modèle capteur inverse : en SLAM probabiliste, le modèle

capteur inverse est l'expression de la distribution de probabilités $p(m_t|x_t,z_t)$, soit la probabilité d'obtenir la carte m_t à l'instant t connaissant la position x_t du robot et l'observation capteur z_t à l'instant t. Dans ce chapitre, on propose l'utilisation de modèles capteurs inverses simples reposant sur une représentation binaire.

4.3.4.1 Grille de perception vidéo

Comme il a été dit dans la section 4.3.1, on suppose qu'une phase de détection d'êtres humains a lieu sur chaque trame de la vidéo. A l'aide d'un recalage d'images, il est possible de faire correspondre un pixel de l'image couleur avec un pixel de l'image de profondeur. Il est donc possible de localiser des pixels détectés comme faisant partie d'un être humain dans l'image de profondeur. Une fois cela fait, on a déjà décrit dans la section 4.3.2.1 le processus permettant de transformer un pixel de l'image de profondeur en point dans le repère de coordonnées du capteur. Ainsi il est possible de replacer une détection vidéo dans une grille. Par ailleurs, chaque impact du scan laser fourni par le télémètre peut être placé dans la grille, comme évoqué en section 4.3.2.2.

Ainsi, le laser et le capteur RGB-D donnent accès à deux types d'information différents pour une cellule donnée :

— Capteur RGB-D : présence d'un humain au sein de la cellule
— Laser : présence d'un obstacle au sein de la cellule

L'exemple jouet est repris dans la figure 4.19 pour montrer la grille de perception vidéo correspondante. On a ici simulé un champ de perception laser égal au champ de vision du capteur RGB-D : 90 °, délimité par les traits rouges sur la figure. Dans la réalité le champ de perception du capteur laser est supérieur à celui du capteur RGB-D.

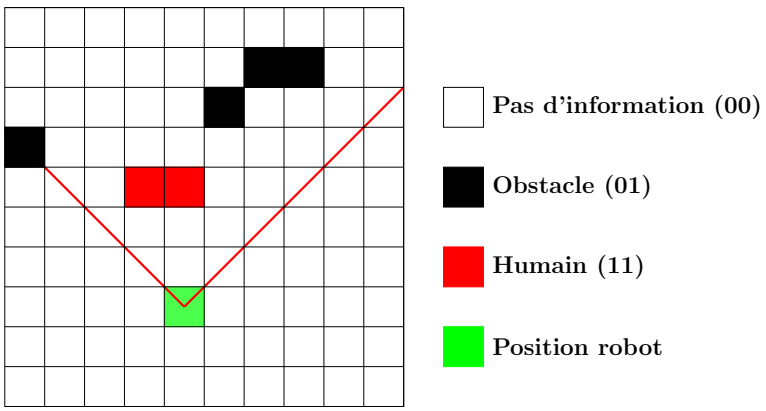


FIGURE 4.19 – Grille de perception vidéo sur l'exemple jouet

On choisit de représenter les différentes valeurs possibles de la grille de perception vidéo sur deux bits. Les 3 types différents pour une cellule sont :

— Pas d'information : 00
— Obstacle : 01

— Humain : 11

Comme on peut le voir, en ce qui concerne la grille de perception vidéo, le robot ne fait pas ici la différence entre des zones non-perçues et des zones libres navigables. Pour calculer les zones libres navigables, il serait nécessaire d'utiliser une méthode telle que le tracé de rayon. Or, on dispose d'une carte d'occupation que l'on utilisera pour la planification de trajectoire, et qui contient déjà l'espace libre navigable. Il est toutefois important d'ajouter les obstacles non-humains à la carte, car ils peuvent représenter un problème lors du déplacement du robot dans le cas où ils ne font pas aussi partie de la grille d'occupation. La grille vidéo contient donc directement les humains et les obstacles perçus.

4.3.4.2 Grille de perception audio

Comme cela a été évoqué dans la section 4.3.1, on fait l'hypothèse que la source sonore, si elle existe, est localisée à chaque instant t d'observation. On a déjà décrit dans le chapitre 3 que localiser précisément une source sonore dans l'espace avec deux microphones est difficile à moindre coût. Il est cependant possible d'obtenir une position angulaire par rapport au robot. Comme on l'a dit, cette estimation angulaire de position est souvent imprécise, la source pouvant se trouver dans une zone angulaire autour de l'angle estimé.

On propose donc de transformer une observation angulaire audio en un cône de localisations possibles de la source dans la grille de perception audio. Toute cellule incluse dans ce cône, dont la largeur dépend de l'incertitude de mesure, est considérée comme étant susceptible de contenir un humain. La grille de perception audio correspondant à l'exemple jouet est illustrée par la figure 4.20.

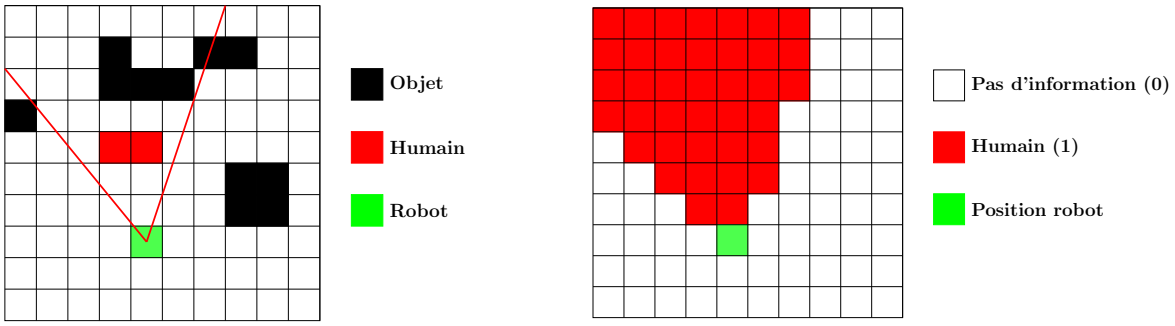


FIGURE 4.20 – Grille de perception audio pour l'exemple jouet. Le cône audio est visible sur l'image de gauche en rouge. La grille de perception audio correspondante est visible sur l'image de droite.

Comme on peut le voir, cette grille ne peut donner qu'un seul type d'information : la possible présence d'humains dans le cône audio. Il est impossible pour la modalité audio de percevoir de l'espace libre navigable ou des obstacles sur le chemin du robot. On a donc une information pour une cellule représentée sur un bit : soit la possible présence d'un humain (1),

soit l'absence d'information (0).

4.3.5 Construction de la grille de perception multimodale

La grille de perception multimodale contient l'ensemble des informations statiques et dynamiques apportées par les différentes modalités pour permettre au robot de prendre une décision quant à sa prochaine destination. On dispose de 5 types d'informations différentes pour une cellule :

- Espace libre navigable, apporté par la grille d'occupation
- Obstacle, apporté par la grille d'occupation (dans le cas d'un objet statique de l'environnement) ou par la grille de perception vidéo (dans le cas d'un objet détecté au cours de la représentation)
- Humain détecté en vidéo, apporté par la grille de perception vidéo
- Humain détecté par l'audio, apporté par la grille de perception audio
- Zone d'intérêt, apporté par la grille sémantique, construite à partir de la grille d'occupation

On propose donc de concaténer l'information au sein d'une même carte sur une valeur représentée par 5 bits. La formulation sera donc de la forme suivante :

bit 4	bit 3	bit 2	bit 1	bit 0
↑	↑	↑	↑	↑
Sémantique	Humain par l'audio	Humain par la vidéo	Obstacle par la vidéo ou l'occupation	Espace libre par l'occupation

(4.14)

Les bits 0 et 1 seront utilisés pour la planification de trajectoire et l'évitement d'obstacles, car ils représentent les obstacles et l'espace navigable, et les bits 2, 3 et 4 seront utilisés par le robot pour prendre une décision sur son déplacement.

Pour donner un exemple, selon cette représentation, une cellule contenant un humain vu et entendu ne se trouvant pas dans une zone d'intérêt de la grille sémantique, sera représenté par la valeur 01110.

4.3.6 Automate de décision

A chaque instant d'observation t, une grille de perception multimodale est construite, qui contient 5 types d'informations. Ces informations doivent être triées par le robot, par ordre de priorité. En effet, le robot ne donnera pas la même importance à une zone d'intérêt de la grille sémantique qu'à une cellule contenant de l'humain perçu par la vidéo. Ainsi, l'ordre de priorité est le suivant :

— Priorité basse : les zones d'intérêt de la grille sémantique représentent le plus faible niveau de priorité pour le robot. Ce sont les endroits qu'il visitera "par défaut" si aucun humain n'est perçu, de quelque manière que ce soit.
— Priorité intermédiaire : le niveau de priorité intermédiaire est donné aux cellules faisant partie du cône audio. Si un humain est entendu, il est intéressant d'essayer de le localiser visuellement.
— Priorité maximale : le plus haut niveau de priorité est donné aux cellules contenant une détection vidéo. En effet, la vision est la modalité la plus importante et la plus robuste pour détecter un être humain.

A partir de cet ordre de priorité, on choisit trois "modes de déplacement", dans lesquels peut se trouver le robot à un instant donné, selon les données perçues : le mode patrouille, le mode audio, et le mode vidéo.

Mode patrouille

C'est le mode "par défaut" du robot. Si aucune information concernant les êtres humains n'est perçue par les capteurs, seule l'information de la grille sémantique peut être utilisée. Le robot élabore son déplacement pour visiter les différentes zones d'intérêt définies dans la grille sémantique en minimisant la distance parcourue : la zone la plus proche de sa position actuelle est choisie comme destination. Lorsqu'une zone de la grille sémantique est visitée, elle est "marquée" comme telle, et est supprimée de la grille sémantique tant que toutes les autres zones de cette grille n'ont pas été visitées. Le mode patrouille sera l'unique mode utilisé si aucun humain ne se trouve dans l'environnement du robot. Il est donc important de définir des zones d'intérêt assez espacées les unes des autres, et qui couvrent l'espace de travail du robot.

Mode audio

Dès qu'une source sonore a été entendue, le robot sort du mode "patrouille" et se met à chercher activement la source entendue. Pour cela, le cône audio ainsi créé est exploré. Des cellules appartenant au cône sont tirées au hasard et sont visitées successivement par le robot, tant que l'humain n'a pas été approché. Si au terme de l'exploration du cône, aucun humain n'est détecté visuellement (pas de cellule marquée "11" dans la grille de perception vidéo), le robot reprend sa patrouille et l'exploration des zones d'intérêt définies par la grille sémantique. Dans le cas où un humain est visuellement détecté, le robot passe alors en mode vidéo.

Mode vidéo

Lorsqu'un ou plusieurs humains sont visuellement détectés, le robot calcule un trajet pour se positionner à un mètre de l'humain le plus proche. Tant que le positionnement n'a pas été effectué, l'humain détecté n'est pas considéré comme approché, et les informations audio et de la grille sémantique ne sont plus prises en compte pour déplacer le robot. Une fois le positionnement effectué, le robot prend une pause de quelques instants, puis reprend son exploration à la recherche d'autres êtres humains.

Automate

Pour gérer le passage entre les différents modes du robot, on parcourt à chaque instant d'observation la grille de perception multimodale pour y chercher l'information prioritaire. Pour cela, la valeur de chaque cellule, codée sur 5 bits, est soumise à trois ET logiques pour trier l'information qu'elle contient. On peut résumer l'automate, illustré par la figure 4.21, de la manière suivante :

— Si au moins une cellule de la grille de perception modale retourne un ET logique non nul avec la valeur 001** (les étoiles représentent un bit dont la valeur n'a pas d'importance), un humain a été visuellement détecté (le bit numéro 2 représentant les détections vidéo) et le robot doit passer en mode vidéo, le mode prioritaire.

— Lorsque le robot se trouve en mode patrouille, si une détection a lieu, quelle qu'elle soit (audio avec la valeur 010** ou vidéo avec la valeur 001**), le robot se place dans le mode correspondant. Tant qu'aucune détection n'a eu lieu (seul le ET logique avec la valeur 100** retourne une valeur non nulle), le robot reste en mode patrouille.

— Lorsqu'un humain a été visuellement détecté, le robot se positionne à un mètre de lui, prend une pause, puis reprend son exploration.

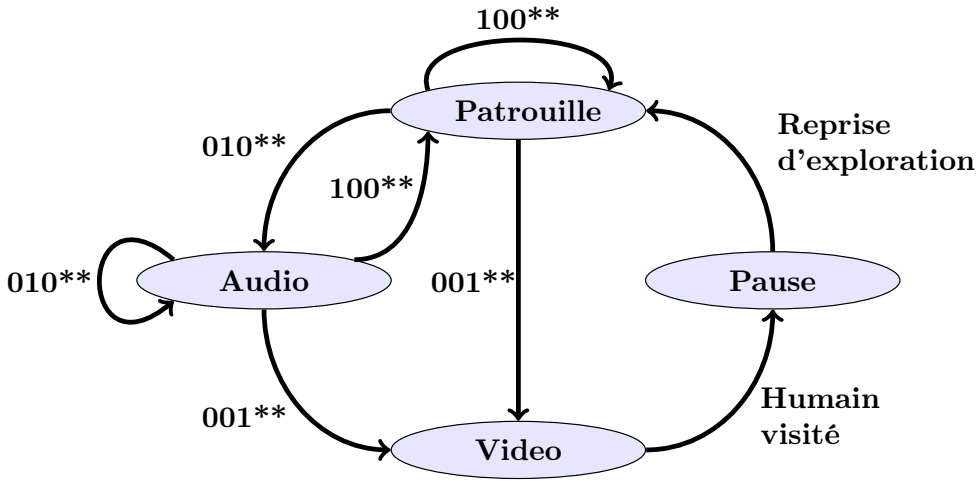
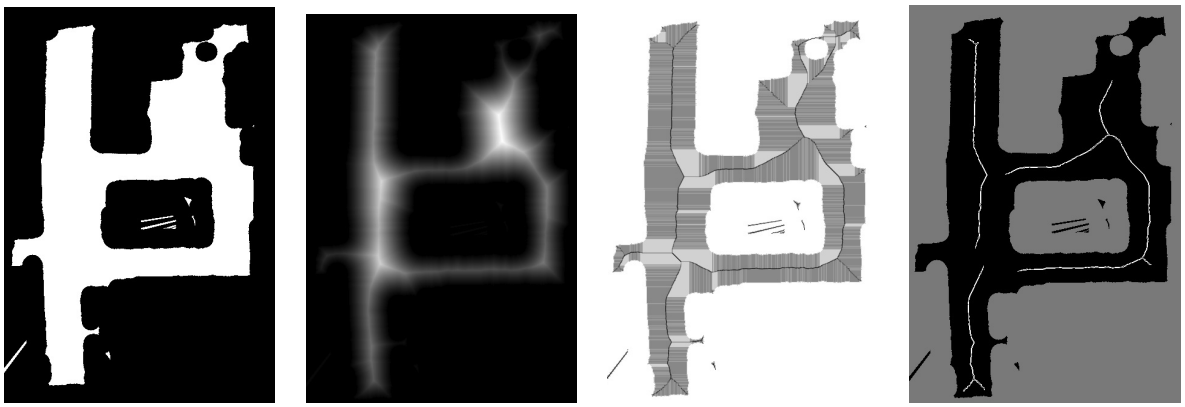


FIGURE 4.21 – Automate pour la décision sur la navigation du robot permettant de gérer les modes de déplacement du robot, de l'ordre de priorité le plus faible (patrouille) à celui le plus grand (vidéo)

4.3.7 Planification de trajectoire

Une fois que le robot a choisi une destination à l'aide des informations perçues, il doit établir un itinéraire, lui permettant de rejoindre cette destination. Pour cela, il est nécessaire de choisir un algorithme de planification de trajectoire, tant au niveau global que pour l'évitement d'obstacles. Pour un robot compagnon, deux considérations naïves semblent être importantes à prendre en compte : il est nécessaire que le robot soit visible lorsqu'il cherche à s'approcher d'un être humain, pour ne pas le prendre par surprise ; et il est nécessaire de rester le plus éloigné possible des obstacles de l'environnement. Les environnements intérieurs sont souvent étroits et encombrés, et il semble donc logique de vouloir que le robot reste le plus éloigné



(a) Carte d'occupation : espace libre en blanc, occupé en noir

(b) Transformée de distance

(c) Gradient

(d) Squelette obtenu

FIGURE 4.22 – Processus de calcul du diagramme de Voronoï généralisé, effectué sur une grille d'occupation d'un appartement réel.

possible des murs et autres objets statiques.

4.3.7.1 Planification globale

Dans le cadre de la navigation robotique classique, la planification de trajectoire globale est celle qui est faite à partir de la grille d'occupation statique. Sans prendre en compte les observations capteurs, le robot élabore le chemin qu'il lui faudra prendre pour atteindre son objectif. Ceci lui permet de créer des trajectoires vers des objectifs en dehors de son champ de perception. En cours de route, cette trajectoire peut être modifiée dans le cas d'évitement d'obstacles ou recalculée entièrement dans le cas d'un obstacle l'empêchant complètement de continuer avec la trajectoire prévue.

Un diagramme de Voronoï généralisé semble adapté aux contraintes fixées de visibilité et de distance aux obstacles. Le diagramme de Voronoï correspond à l'ensemble des points les plus éloignés des objets de la grille d'occupation. Il se calcule en 3 étapes, illustrées par la figure 4.22.

— Calcul de la transformée de distance : la transformée de distance consiste à assigner à chaque cellule de la grille une valeur correspondant à sa distance en nombre de cellules au plus proche obstacle. On utilise une technique dite du feu de prairie : on initialise l'algorithme sur chacune des cellules contenant un obstacle, voisine d'une cellule libre, à 0. Chaque cellule libre, voisine des cellules initialisées, reçoit une valeur de 1, et les distances sont propagées jusqu'à avoir assigné une valeur à chacune des cellules de la carte.
— Calcul du gradient : le gradient est calculé à chaque cellule de la transformée de distance.

— Une descente de gradient est calculée pour obtenir les minimum de gradient dans la grille, qui correspondent à l'ensemble des points les plus éloignés des obstacles de la grille.

Une fois le diagramme obtenu, calculer un chemin devient plus aisé. On considère que chaque cellule appartenant au diagramme de Voronoï est un noeud d'un graphe, et que ses voisins directs dans le diagramme sont ses noeuds voisins dans le graphe avec une arête ayant un poids de 1. On peut donc utiliser un algorithme de plus court chemin pour se déplacer sur le diagramme de Voronoï.

Ainsi, pour se déplacer d'une cellule m^i à une cellule m^l de la grille d'occupation, pour le robot la stratégie est la suivante : on recherche la cellule m^j (resp. m^k) du diagramme de Voronoï la plus proche de m^i (resp. m^l). Le chemin final suivi sera donc $\{m^i, m^j, m^k, m^l\}$.

Pour permettre au robot de ne pas avoir à calculer plusieurs fois le diagramme de Voronoï, on choisit aussi ici de le calculer directement sur la grille d'occupation, en amont de la recherche d'humains par le robot. Toutefois, des obstacles n'apparaissant pas dans la grille d'occupation peuvent apparaître au cours du déplacement (humain, mobilier déplacé...) dans la grille de perception multimodale (bit numéro 1 de la représentation), empêchant complètement le passage du robot et "coupant" ainsi le diagramme de Voronoï. Calculer à nouveau l'intégralité du diagramme est coûteux, et non nécessaire. [LSB13] propose une technique de mise à jour rapide du diagramme de Voronoï, reposant sur le fait de ne recalculer que partiellement la transformée de distance, en gardant en mémoire pour chaque cellule, en plus de sa distance à l'obstacle le plus proche, la localisation de cet obstacle. Cela permet, lors du "feu de prairie", de sélectionner les cellules nécessitant une mise à jour de leur distance, et de ne pas parcourir intégralement la carte.

Dans le cas où l'obstacle apparu dans la grille de perception multimodale peut être contourné, c'est l'algorithme d'évitement d'obstacles qui intervient.

Evitement d'obstacles et exécution de la trajectoire

Pour l'évitement d'obstacles, il existe déjà de nombreux planificateurs performants de la littérature, dont le code est disponible sur internet. Nous avons choisi une approche de type DWA [FBT97].

Cette méthode repose sur le fait que le mouvement d'un robot à un instant donné peut être décrit par un couple (v,w), correspondant respectivement à une vitesse de translation et une vitesse de rotation. L'objectif est de permettre au robot à chaque instant de prévoir une trajectoire évitant les obstacles de son environnement. L'algorithme fonctionne selon les étapes suivantes :

— Création d'une fenêtre de recherche autour du robot. Cette fenêtre est centrée sur la position du robot et dépend de sa vitesse et son accélération à l'instant considéré.
— Tirage de trajectoires dans cette fenêtre de recherche : un ensemble de trajectoires possibles du robot dans la fenêtre dynamique formée est tiré. Une trajectoire correspond

à une prédiction de positions successives au sein de la fenêtre.
— Assignation des scores à chacune des trajectoires. Chaque trajectoire reçoit un score dépendant de trois facteurs : la direction de la trajectoire (une trajectoire rapprochant le robot de son but sera favorisée), la distance des obstacles de l'environnement à cette trajectoire (plus la distance du plus proche obstacle à la trajectoire est petite, meilleur sera le score de cette trajectoire), et la vitesse possible sur cette trajectoire (les trajectoires droites permettant des vitesses plus grandes seront favorisées).
— Sélection de la trajectoire ayant reçu le meilleur score : la trajectoire choisie sera donc celle qui permet au robot de se mouvoir le plus efficacement possible dans la direction de son but.

4.4 Résultats

Le système présenté dans ce chapitre a été implémenté sur un robot réel. On propose de diviser cette partie de la manière suivante : le robot est présenté dans la section 4.4.1, puis l'architecture logicielle est brièvement détaillée dans la section 4.4.2. Les conditions d'expérimentation et les scénarios proposés sont décrits dans la section 4.4.3 puis illustrés dans la section 4.4.4

4.4.1 Le robot Q.bo

Le robot utilisé est Q.bo (figure 4.23), produit par la compagnie espagnole TheCorpora^{TM} [1]. Ce robot, haut de 45 centimètres, dispose d'un corps rond, surmonté d'une tête mobile, équipé de deux caméras faisant office d'yeux, ainsi que deux microphones sur les bords de la tête. Ce robot, open-source (aussi bien en terme matériel que logiciel), est entièrement programmable.

Pour avoir accès à l'information de profondeur, Q.bo a d'abord été équipé d'un capteur RGB-D Asus Xtion Pro Live, fixé au sommet de sa tête. Ce capteur a l'avantage de disposer d'une paire de microphones, chacun placé à une extrémité. Ce sont ces microphones qui sont utilisés lors de l'expérimentation : ils sont plus éloignés du corps du robot, et ainsi moins sensibles au bruit produit par le robot lors de ses déplacements.

Les caractéristiques du capteur RGB-D Xtion sont les suivantes :

— Champ de perception : 50 ° horizontal, 45 ° vertical
— Résolution des images RGB et de profondeur : 640x480 pixels
— Fréquence de rafraîchissement : 30 Hz
— Profondeur perceptible : de 50 cm à 4m environ

Pour obtenir une information de distance plus robuste, nous avons équipé le robot d'un capteur laser Hokuyo URG-04LX-UG01. Ce type de capteur est plus précis qu'un capteur

1. www.thecorpora.com

FIGURE 4.23 – Le robot Q.bo, produit par TheCorpora

RGB-D pour la profondeur, et le champ de perception est supérieur à celui du capteur Xtion. Les caractéristiques du laser utilisé sont les suivantes :

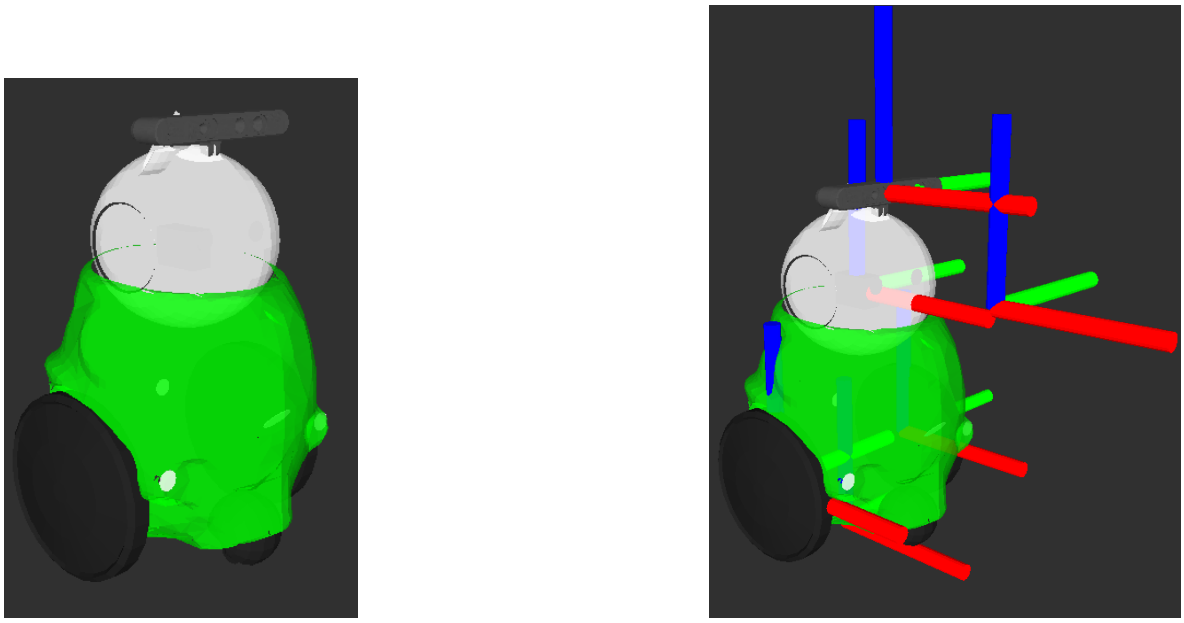
— Champ de perception angulaire : 210 °
— Résolution angulaire : 0.4 ° (525 rayons par scan laser).
— Fréquence de rafraîchissement : 10 Hz
— Profondeur perceptible : de 2cm à 5m environ

Le capteur RGB-D est donc utilisé uniquement pour la détection et le replacement d'humains dans la grille. La grille de perception vidéo se compose ainsi du scan laser donné par le capteur Hokuyo, et d'éventuelles détections d'humains données par le capteur RGB-D.

Concernant les actionneurs, Q.bo dispose de deux roues motrices et d'une roue folle. Chaque roue motrice est équipée d'un encodeur magnétique permettant d'estimer le déplacement du robot. La tête du robot peut tourner en lacet ("yaw" en anglais) et en tangage ("pitch"). Finalement, en guise de ressources de calcul, le robot dispose d'un processeur Intel i3-2120T, et de 4Gb de RAM DDR3.

Pour que le robot soit capable de transformer les observations produites d'un repère capteur à un autre, il est nécessaire de construire un arbre cinématique : c'est dans cet arbre que sont définis les différents repères qui composent le robot. Nous avons construit un arbre cinématique pour le robot Q.bo, dont certains repères sont visibles dans la figure 4.24.

Comme on peut le voir, nous avons défini un repère pour chacune des roues, pour la base du robot, sa tête, et pour le capteur RGB-D. Le dernier repère, situé à l'avant du robot,



(a) Représentation 3D du robot

(b) Représentation des repères principaux du robot

FIGURE 4.24 – Représentation d'une partie de l'arbre cinématique du robot

est celui du capteur laser. Cependant, à ce stade, la représentation 3D du laser est encore indisponible. C'est pourquoi il n'apparaît pas visuellement dans la figure. L'arbre cinématique complet du robot est décrit dans l'annexe A.

4.4.2 Architecture logicielle

L'architecture logicielle a été réalisée à l'aide du middleware "Robot Operating System" (ROS). Ce logiciel, spécialement prévu pour la conception de programmes informatiques pour la robotique, fournit un ensemble de structures et de bibliothèques agissant comme interface entre les capteurs, les actionneurs, et les programmes de décision du robot. ROS permet l'utilisation d'architectures modulaires en donnant accès à différents éléments dont les principaux sont les suivants :

— Un **noeud** est un morceau de code ayant une fonction précise, qui peut être exécuté par le robot.
— Un **topic** est un canal de communication entre deux noeuds. Il permet à un noeud de préciser ses sorties. De la même manière, il permet à un noeud d'utiliser les sorties d'un autre noeud comme ses propres entrées.
— Un **message** est une structure de transport d'information entre deux noeuds. Un message est publié sur un topic, et permet le transfert effectif d'informations importantes.

Cette architecture en noeuds, et la quantité de messages différents existant nativement dans ROS, rendent l'intégration de nouveaux modules aisée. L'architecture logicielle du système présenté dans ce chapitre est illustrée de manière simplifiée par la figure 4.25 (l'architecture logicielle complète est donnée dans l'annexe B) : on peut y voir les noeuds principaux (les boîtes), certains topics (les flèches), et le type de messages circulant sur ces topics (annotation des flèches). Pour alléger la figure, certains noeuds secondaires de l'architecture réelle ont été retirés, ainsi que des topics reliant certains noeuds.
Sur cette figure, les noeuds comportant le préfixe "qbo_" sont ceux que nous avons développés spécialement pour le système présenté dans ce chapitre. Les autres sont des noeuds génériques ROS, configurés et utilisés tels quels. On peut diviser l'architecture en trois grandes parties :

— "Capteurs" : ces noeuds transmettent les données capteurs à tout le système
— "Traitement de données" : ces noeuds traitent les données capteurs et les mettent en forme pour la construction de grilles
— "Navigation" : ces noeuds construisent les grilles, choisissent l'objectif du robot et lui envoient les commandes pour réaliser la trajectoire choisie

Les noeuds de la partie "capteurs" servant uniquement à extraire les données des différents capteurs dont le robot est équipé, on choisit de détailler les noeuds des parties "Traitement de données" et "Navigation" dans les tableaux 4.1 et 4.2 respectivement. Comme on peut le voir, le noeud principal, qbo_perceptionmultimodale, ne représente qu'une petite partie de l'architecture logicielle globale.

4.4.3 Conditions d'expérimentation

4.4.3.1 Salle d'expérimentation

Les manipulations ont eu lieu au sein d'une salle de classe transformée en espace de vie (figure 4.26). Les éléments statiques (donc faisant partie de la grille d'occupation) présents sont : une table, un espace de discussion avec une table basse et deux fauteuils, et un petit mur élaboré à l'aide de cartons.

Les dimensions de la salle sont de 8m x 7.3m. La grille d'occupation correspondant à cette pièce est illustrée par la figure 4.27. Le blanc représente l'espace libre navigable, le gris l'espace non exploré, et le noir les obstacles statiques de la pièce. La résolution de la grille d'occupation est de 10 cm (chaque cellule de la grille est un carré de 10x10 cm).

Le robot mesurant 45 centimètres de haut, les obstacles statiques de l'environnement peuvent très facilement obstruer la vision du robot (figure 4.28). Pour l'empêcher de naviguer dans des endroits "dangereux" (sous les tables), des nappes ont été posées sur les obstacles trop hauts pour qu'ils soient détectés à tout moment.

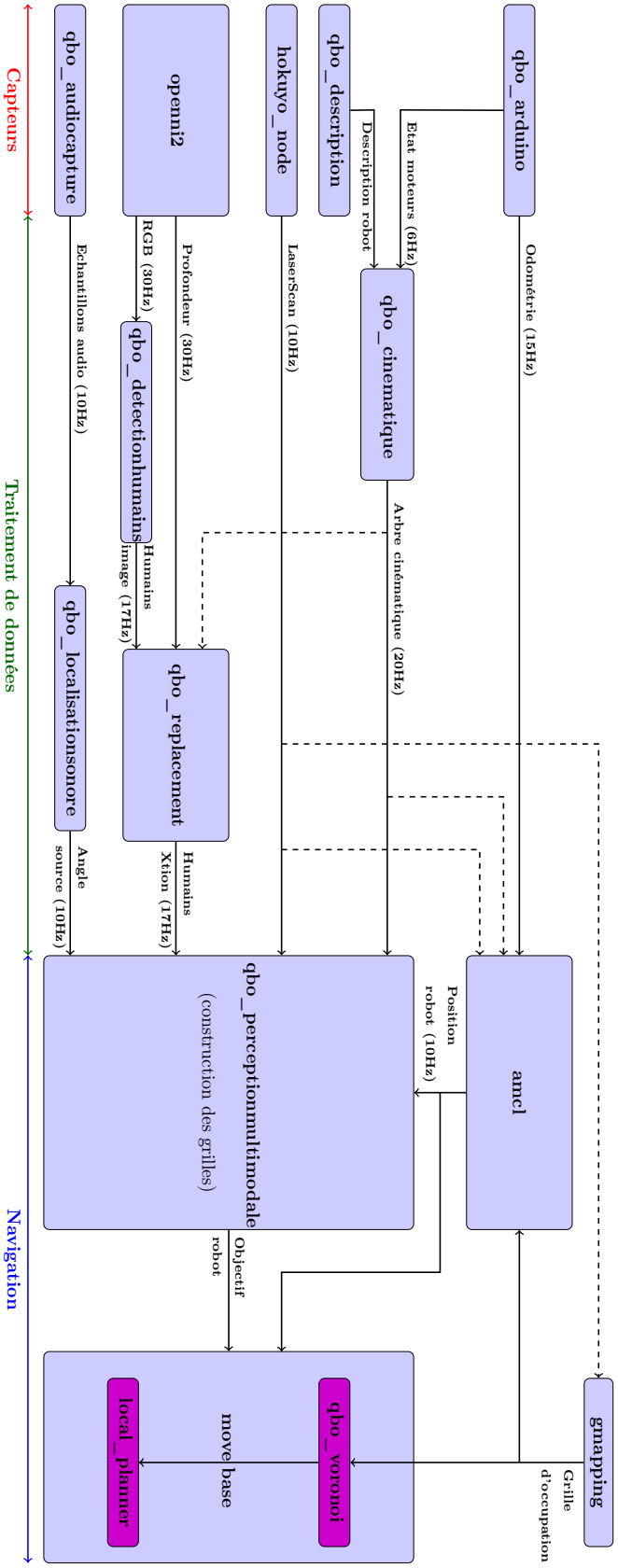


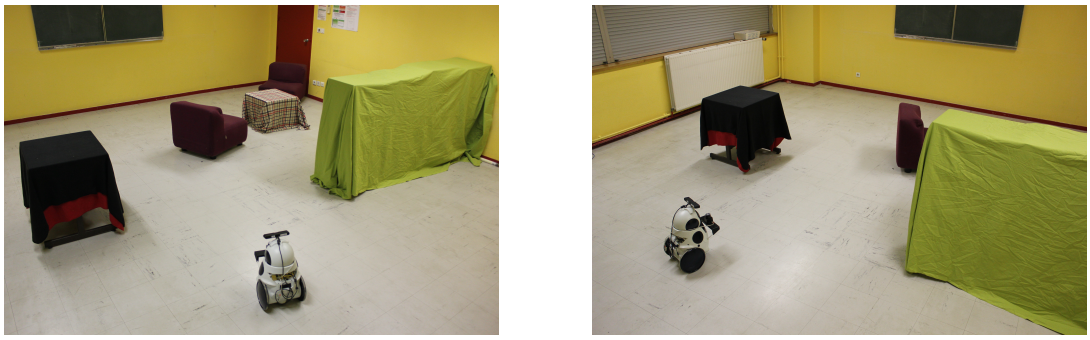
FIGURE 4.25 – Architecture logicielle : Noeuds, topics et messages

Noeud	Entrées	Sorties	Description
qbo_cinematique	Description robot Etats moteurs tête	Arbre cinématique	Un robot est représenté par un ensemble de systèmes de coordonnées, dans lesquels les différentes données capteurs sont acquises. Il est nécessaire de pouvoir transformer une observation d'un repère à un autre à chaque instant. Par exemple, on doit pouvoir replacer une observation faite dans le repère du capteur RGB-D Xtion dans le repère de la base du robot. Le noeud qbo_cinematique publie l'arbre des différents repères fixes du robot, appelé arbre cinématique.
qbo_detectionhumains	Image RGB	Humains dans l'image	Détection d'humains dans l'image, avant le replacement dans la grille de perception vidéo
qbo_replacement	Humains dans l'image Image de profondeur Arbre cinématique	Humains dans le repère du capteur RGB-D Asus Xtion	Replacement des observations dans le repère du capteur RGB-D, à l'aide de la méthode décrite dans la section 4.3.2.1. Ceci permet au robot d'obtenir un point dans son environnement en trois dimensions.
qbo_localisationsonore	Échantillons audio	Angle auquel la source a été entendue (si une source a été entendue)	Noeud de localisation de source sonore : si une source a été entendue, l'angle est donné, dans le repère du capteur RGB-D (on utilise les microphones de ce capteur). Sinon, un angle infini est donné, pour signifier au module de construction de grille de ne pas prendre en compte l'information sonore.

TABLE 4.1 – Description des différents noeuds de la partie "Traitement de données" de l'architecture logicielle

Noeud	Entrées	Sorties	Description
amcl	Scan laser Grille d'occupation Odométrie	Position du robot dans la grille d'occupation	AMCL est l'acronyme de "A Monte Carlo Localization". Noeud de localisation du robot, à l'aide de l'algorithme de Monte Carlo évoqué dans la section 4.2.4. Plus d'informations : Site amcl
gmapping	Scan laser Odométrie (non apparent sur la figure) Arbre cinématique (non apparent)	Grille d'occupation de l'environnement	gmapping utilise l'algorithme de [GSB07] évoqué dans la section 4.2.3 pour créer une grille d'occupation de l'environnement. Plus d'informations : Site gmapping
qbo_perceptionmultimodale	Scan Laser Humains repère Xtion Angle source sonore Arbre cinématique	Objectif robot	Ce noeud est le coeur du système décrit dans ce chapitre. Il prend en entrée des données de profondeur, de présence d'humains (audio ou vidéo), et construit les grilles décrites en section 4.3. Une fois ces grilles construites, la décision est prise quant à la prochaine destination du robot. Dans le cas où le robot est déjà en déplacement vers une destination, aucun objectif n'est envoyé en sortie.
move_base	Objectif robot Position robot Grille d'occupation Scan laser (non apparent) Arbre cinématique (non apparent)	Commande robot (non apparent)	Ce noeud fait partie intégrante de l'architecture de navigation native de ROS : il contient toutes les structures et algorithmes liés à la planification de trajectoire et leur exécution. Les deux pièces principales sont le planificateur de trajectoire globale (**qbo_voronoi**, détaillé plus loin dans la section), et le planificateur de trajectoire locale (local_planner). Plus d'informations : Site move_base

TABLE 4.2 – Description des différents noeuds de la partie "Navigation" de l'architecture logicielle



(a) Vue 1 (b) Vue 2

FIGURE 4.26 – Salle d'expérimentation robotique

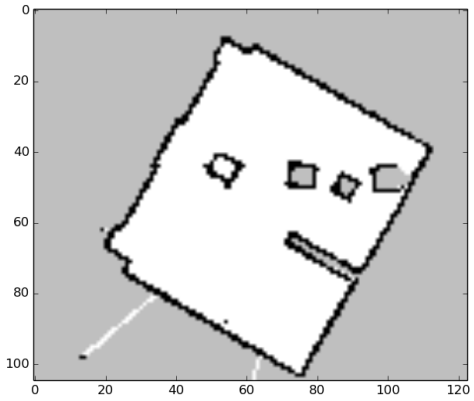
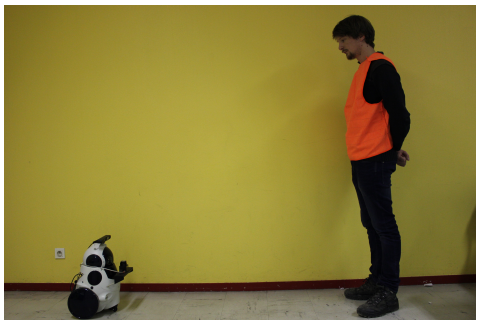


FIGURE 4.27 – Grille d'occupation représentant la salle d'expérimentation (10 cm de résolution)

FIGURE 4.28 – Le robot Q.bo à côté d'un fauteuil bas : si le robot est proche du fauteuil, celui-ci obstruera entièrement sa perception visuelle

(a) Exemple de gilet porté par un humain



(b) Cas où le robot ne pourra pas détecter l'humain

FIGURE 4.29 – Conditions d'expérimentation pour la vidéo

4.4.3.2 Détecteurs utilisés

Pour illustrer le fonctionnement du système et ménager les ressources de calcul du système, des détecteurs simples ont été choisis :

- **Détecteur vidéo :** pour localiser aisément l'humain, les personnes présentes dans l'environnement du robot sont équipées de gilets de sécurité rouge, à la couleur vive (figure 4.29 (a)). Cette couleur est apprise à l'aide du classifieur bayésien naïf utilisé dans le chapitre 3. Une détection est considérée comme telle si la zone détectée contient assez de pixels. A nouveau la taille du robot peut poser ici problème : si jamais le robot se trouve trop proche de l'humain debout (figure 4.29 (b)), il ne pourra pas le détecter. La capacité du robot à tourner la tête en tangage ("pitch") n'est pas utilisée : la transformation entre le repère de la tête et celui du corps du robot est trop lente pour pouvoir transformer une observation faite par le capteur RGB-D dans le repère du robot en temps réel.
- **Localisation de source sonore :** la méthode de localisation de source sonore utilisée est la même que celle qui est présentée dans le chapitre 3. Comme on l'a évoqué, la localisation sonore n'est pas robuste, particulièrement en cas de mouvement (du robot où des éléments de la scène). Pour améliorer la localisation, nous avons choisi de créer notre propre source sonore : un bruit blanc. Le bruit blanc étant décorrélé, on minimise ainsi l'impact des trajets secondaires et du manque d'énergie. Ainsi, chaque humain est équipé d'un générateur sonore et d'un petit haut-parleur qui lui permet de simuler à volonté une prise de parole.

4.4.3.3 Scénarios proposés

Pour illustrer le fonctionnement du système proposé, on propose la réalisation de deux scénarios simples :

— **Expérience 1 : patrouille du robot.** Aucun humain n'est présent dans l'environnement du robot. Le robot entame donc sa patrouille, et parcourt les points de la grille sémantique sans voir personne, pour finalement s'immobiliser, ayant exploré la pièce.
— **Expérience 2 : humain attirant l'attention du robot en patrouille.** Le robot commence à patrouiller, hors du champ de vision de l'humain. Lorsque ce dernier l'aperçoit, il l'interpelle (avec du son) pour attirer son attention. Le robot se tourne donc vers la source de son entendue, puis ayant détecté l'humain visuellement, choisit son chemin pour aller se positionner à environ 1 mètre de lui.

4.4.4 Réalisation des scénarios

4.4.4.1 Diagramme de Voronoï

Le diagramme de Voronoï est calculé par le noeud ROS qbo_voronoi présenté dans la figure 4.25. L'interface de navigation de ROS (navigation stack en anglais) dispose déjà de planificateurs de trajectoire globale. Ils sont utilisables sous forme de module d'extension (plugin), et configurables lors de l'exécution du noeud "move_base". Au moment de la création de ce système, cependant, aucun planificateur global reposant sur un diagramme de Voronoï n'était disponible.

Pour faciliter l'intégration du diagramme de Voronoï à l'interface de navigation ROS, nous avons choisi de coder qbo_voronoi comme un module d'extension de l'interface de navigation. Ainsi, tout robot configuré pour utiliser l'interface de navigation ROS peut utiliser qbo_voronoi. L'algorithme de création du diagramme de Voronoï repose sur plusieurs étapes :

— Seuillage de la grille d'occupation : tout cellule ayant reçu une probabilité d'occupation supérieure à un seuil arbitraire est considérée comme occupée (figure 4.30 (a)).
— Dilatation des objets statiques de la grille : pour minimiser au maximum les risques de collisions du robot avec les objets de son environnement, chaque objet de la carte est dilaté (figure 4.30 (b)).
— Calcul de la transformée de distance : la transformée de distance est calculée de la manière décrite dans la section 4.3.7 : tout d'abord, les cellules libres adjacentes à des cellules occupées sont ajoutées à une queue informatique, puis leur distance est mise à jour. Ce processus a lieu tant qu'il existe des cellules adjacentes dont la distance n'a pas été mise à jour. Deux étapes du calcul de la transformée de distance sont illustrées par les figures 4.30 (c) et (d). Dans la figure 4.30 (c) les éléments mis en queue sont tous ceux qui sont adjacents à un obstacle (pixels en blanc). Lors de la huitième étape, les pixels qui étaient présents dans la queue lors de l'étape précédente ont une distance fixée (gris), et les pixels adjacents libres sont mis en queue à leur tour (blanc). Au terme des itérations (toutes les distances ont été mises à jour), on obtient la transformée de distance (figure 4.30 (e)).
— Pour obtenir le diagramme de Voronoï final, on choisit de parcourir la transformée de distance et de garder toutes les cellules qui remplissent la condition suivante : pour

une cellule donnée, si parmi ses 8 voisins, trois pixels ou moins ont une valeur de transformée de distance supérieure ou égale à la cellule considérée, alors elle fait partie du diagramme de Voronoï. Le résultat est donné en figure 4.30 (f).

Comme on peut le voir, le diagramme de Voronoï donne beaucoup de branches possibles. Ceci est dû à la méthode de sélection des cellules de la transformée de distance, et représente un problème connu de la littérature. Dans notre cas, obtenir de nombreuses branches n'est pas un inconvénient car cela permet au robot de s'approcher facilement d'un humain détecté. De plus, les trajectoires calculées sur ce diagramme, a priori peu naturelles, seront rendues plus naturelles par le planificateur local, qui permettra d'arrondir des trajectoires trop carrées, ou de prendre des raccourcis si aucun obstacle n'est présent sur la trajectoire.

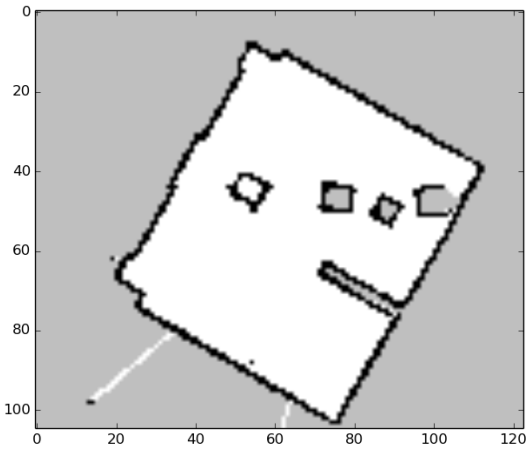
4.4.4.2 Expérience 1 : patrouille du robot

Dans cette expérience, comme décrit dans la section précédente, aucun humain n'est présent dans la scène. Le comportement attendu pour le robot est donc de rester en mode "patrouille" jusqu'à avoir visité tous les points de la grille sémantique. Pour cette expérience, 2 points sémantiques ont été arbitrairement positionnés dans la grille d'occupation (figure 4.31). Ce sont les positions qui donnent le plus de chance au robot d'observer la scène.

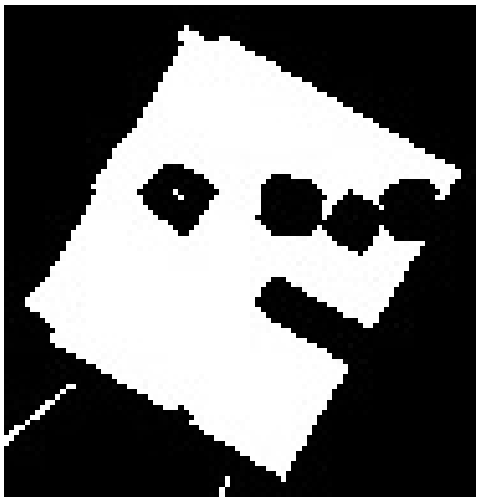
Pour illustrer le scénario, on choisit de montrer à différents instants : le diagramme de Voronoï, avec la position du robot et son objectif ; une photo correspondante de la scène en cours ; la grille de perception vidéo, et la grille de perception audio. Le résultat est montré par les figures 4.32 et 4.33. Concernant le diagramme de Voronoï sur ces figures : les positions sont données par le disque de couleurs, et les orientations par les flèches. Pour ce qui est des grilles de perception : l'espace libre navigable est représenté en vert, les obstacles en rouge, et les humains détectés en bleu.

— **t = 0s : première perception.** Le robot n'a pas perçu d'humain dans son environnement. Il commence donc par chercher le point de la grille sémantique le plus proche de sa position actuelle pour aller le visiter (figure 4.32 (a)). Seul le scan laser est présent dans la grille de perception vidéo (figure 4.33 (a)), et la grille audio est vide, aucune source sonore n'ayant été entendue (figure 4.33 (b)). Sur la grille de perception vidéo, l'espace libre navigable perçu par le robot a été explicitement représenté pour faciliter la lecture des résultats.
— **t = 18s : le robot a visité son premier objectif dans la grille sémantique.** N'ayant toujours pas perçu d'humain (figures 4.33 (c) et (d)), il choisit le prochain point de la grille sémantique le plus proche comme objectif (figure 4.32 (c))
— **t = 66s : le robot a fini de visiter la grille sémantique.** N'ayant pas perçu d'humain au cours de sa patrouille, le robot s'arrête en position finale.

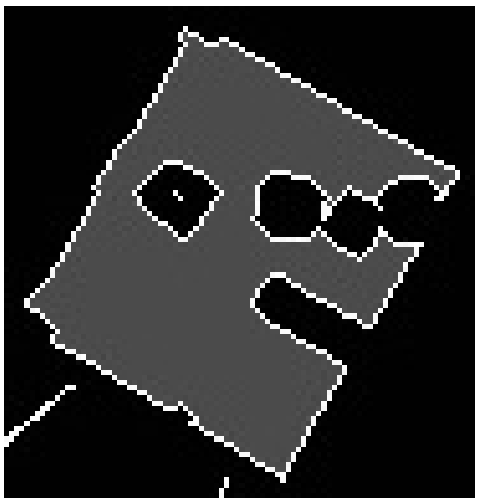
Le robot effectue bien sa patrouille comme prévu, avec le comportement attendu.



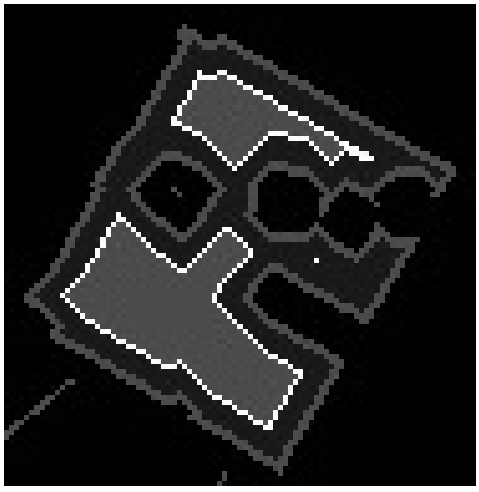
(a) Carte d'occupation seuillée



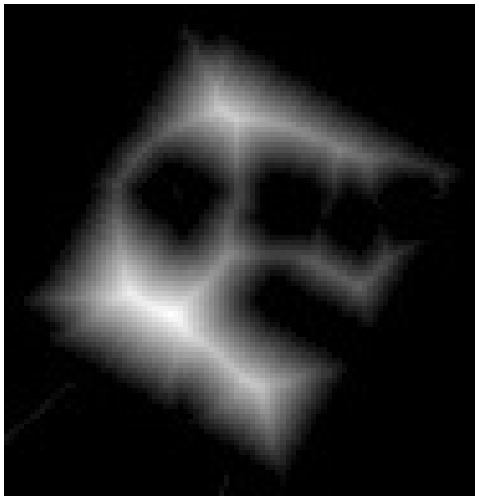
(b) Carte d'occupation dilatée



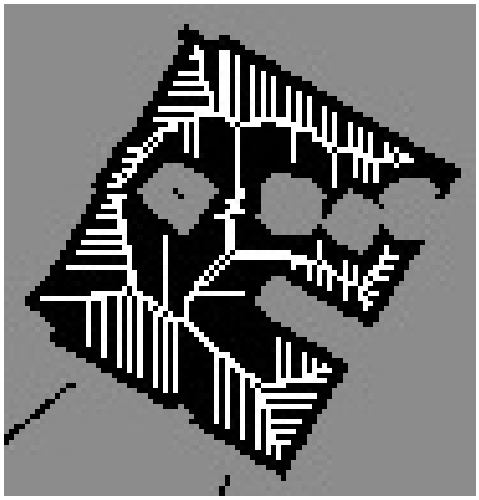
(c) Création de la carte de distance : première itération



(d) Création de la carte de distance : huitième itération



(e) Transformée de distance normalisée



(f) Diagramme de Voronoï final

FIGURE 4.30 – Processus de construction du diagramme de Voronoï de la salle d'expérimentation

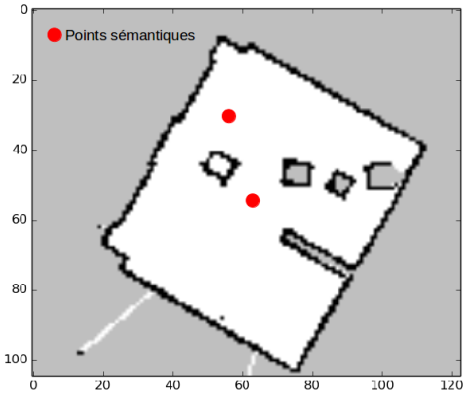
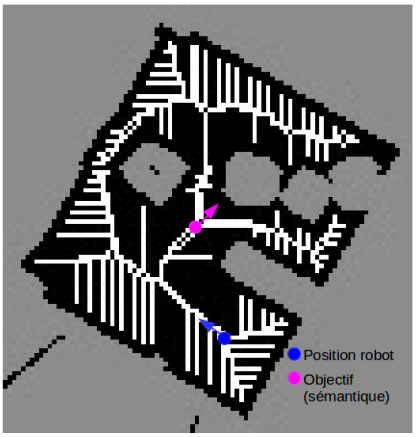


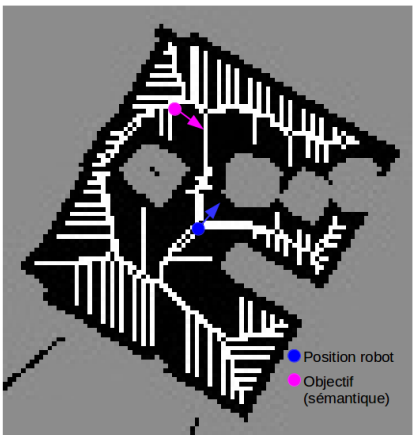
FIGURE 4.31 – Points d'intérêt dans la grille sémantique

Diagramme de Voronoï Photo de la scène



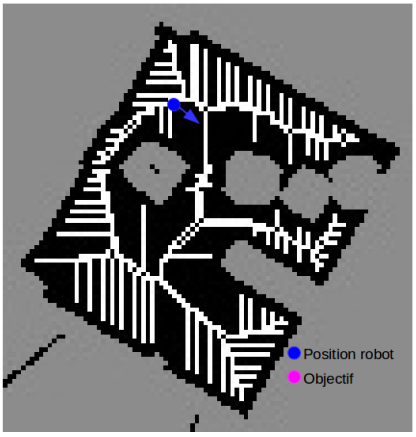
(a) t = 0s

(b) t = 0s



(c) t = 18s

(d) t = 18s



(e) t = 66s

(f) t = 66s

FIGURE 4.32 – Expérience 1 : Diagramme de Voronoï et photo de la scène à 3 instants d'observation

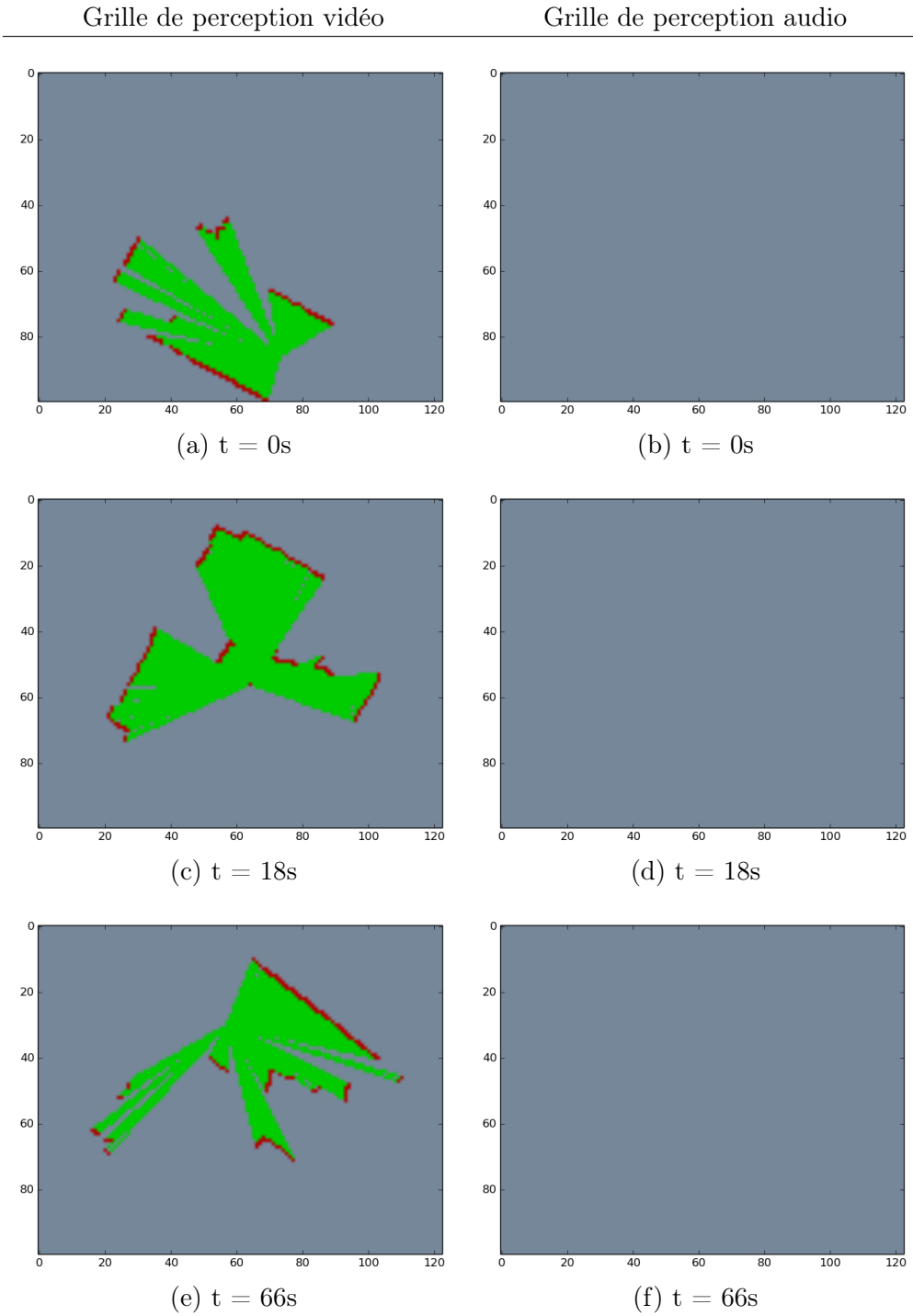


FIGURE 4.33 – Expérience 1 : grille de perception vidéo et grille de perception audio. Les obstacles sont représentés en rouge, l'espace libre en vert et les humains détectés en bleu

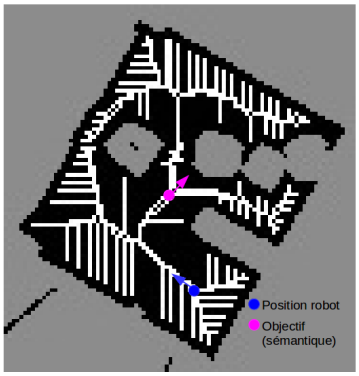
4.4.4.3 Expérience 2 : humain attirant l'attention du robot en patrouille

Le début de ce scénario est le même que dans cette expérience précédente. Toutefois, un humain est présent dans la scène. Il se trouve hors du champ de perception du robot à t=0, tout comme le robot se trouve hors du champ de perception de l'humain. Dès que l'humain aperçoit le robot il "l'interpelle" à l'aide d'un signal sonore. Les résultats aux instants clés du scénario sont présentés dans les figures 4.34 et 4.35.

- **t = 0s : première perception**. La première perception est similaire à celle du premier scénario : le robot n'ayant pas encore perçu d'être humain, il choisit le point de la grille sémantique le plus proche comme objectif.
- **t = 10s : l'humain interpelle le robot**. A ce moment là, le robot entend pour la première fois une source sonore sur sa droite (figure 4.35 (d)). Le robot n'ayant pas détecté visuellement d'être humain (figure 4.35 (c)), il entre dans le mode "audio" et a pour objectif de se tourner la source sonore entendue (figure 4.34 (c)).
- **t = 20s : le robot détecte l'humain visuellement.** Au cours de sa rotation, le robot détecte l'humain à l'aide du capteur RGB-D (figure 4.35 (e)). Il entre donc dans le mode "vidéo" et cherche le point du diagramme le Voronoï le plus proche de l'être humain qu'il souhaite approcher. Une fois ce point trouvé (figure 4.35 (d)), il entame son mouvement pour aller se positionner face à l'être humain. Toute autre information, telle que le cône audio (figure 4.35 (f)) n'est plus considérée. Il est intéressant de noter que dans ce cas, le capteur RGB-D a pu détecter l'humain à un endroit non perceptible par le laser. Ceci est dû au fait que le capteur RGB-D se trouve plus haut que le capteur laser.
- **t = 56s : le robot est positionné.** Le robot a suivi sa trajectoire avec succès et s'est arrêté face à l'être humain, en position d'interagir avec lui. L'être humain continue d'être détecté par le capteur RGB-D (figure 4.35 (g)). L'humain est désormais silencieux (figure 4.35 (h))

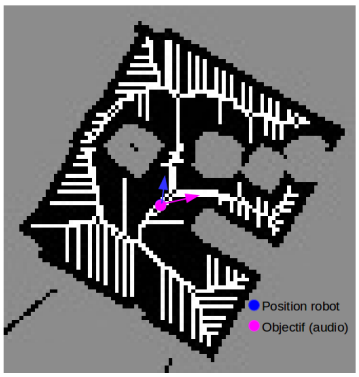
Le robot change ainsi de mode de la manière attendue, réagissant aux différents types d'informations perçus.

Diagramme de Voronoï	Photo de la scène



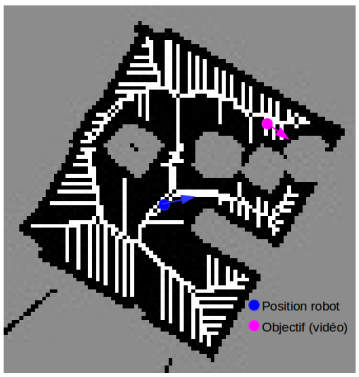
(a) t = 0s

(b) t = 0s



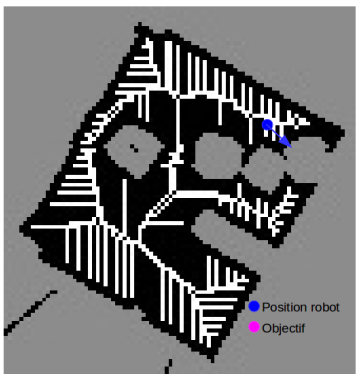
(c) t = 10s

(d) t = 10s



(e) t = 20s

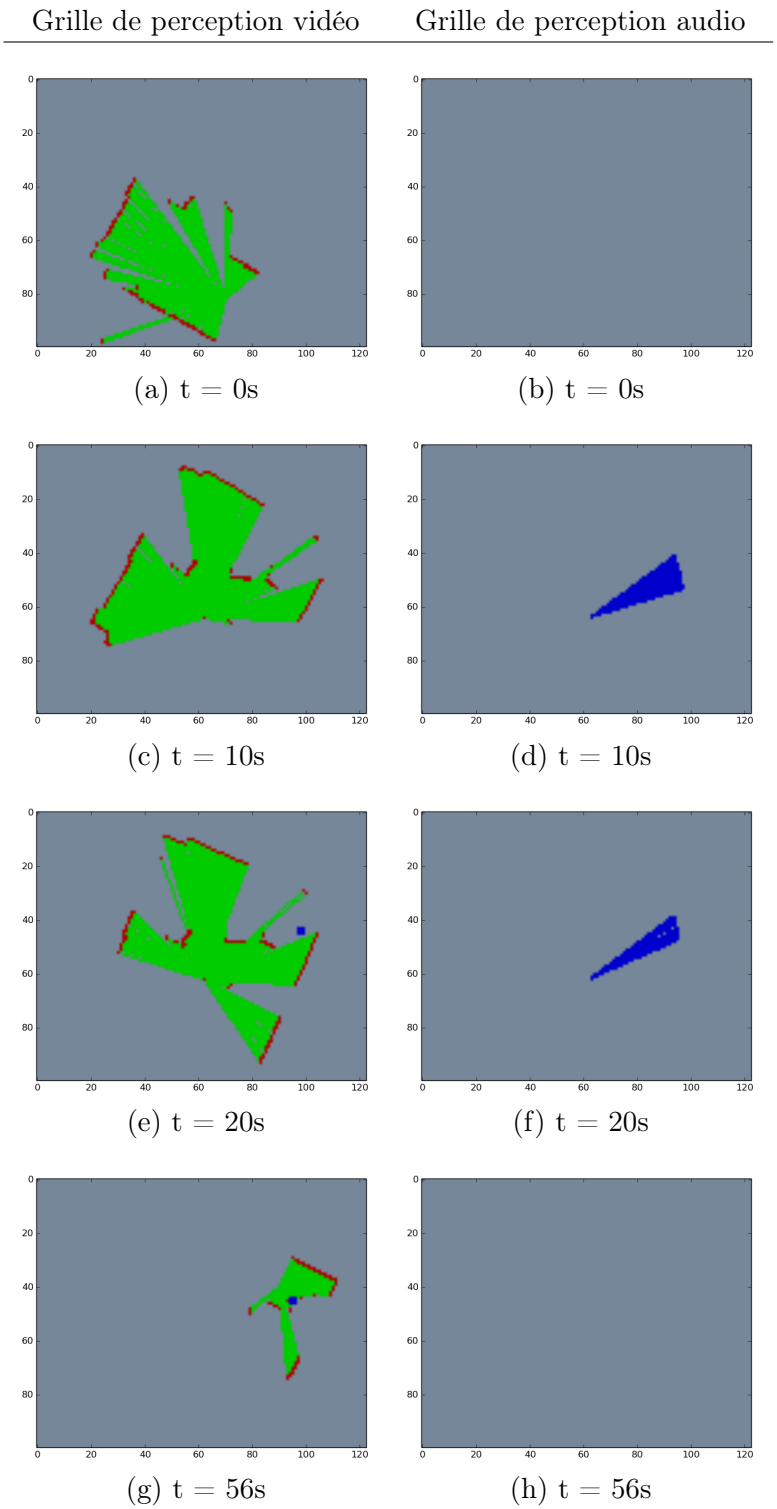
(f) t = 20s



(g) t = 56s

(h) t = 56s

FIGURE 4.34 – Expérience 2 : Diagramme de Voronoï et photo de la scène à 4 instants d'observation



(a) t = 0s (b) t = 0s

(c) t = 10s (d) t = 10s

(e) t = 20s (f) t = 20s

(g) t = 56s (h) t = 56s

FIGURE 4.35 – Expérience 2 : grille de perception vidéo et grille de perception audio

4.5 Conclusion

Dans ce chapitre, un système de navigation autonome, reposant sur une fusion multimodale, a été présenté. Il inclut les données des capteurs suivants : laser, RGB-D et audio. Une grille de perception est construite pour chaque capteur et l'information est traitée par ordre de priorité à l'aide d'un automate. La planification de trajectoire est effectuée à l'aide d'un diagramme de Voronoï, permettant au robot de rester aussi éloigné que possible des obstacles de l'environnement. Ce système présente plusieurs avantages :

— Il repose sur des modèles capteurs inverses simples et peu coûteux en ressources de calcul.
— Le système est entièrement intégré dans l'interface de navigation de ROS, et utilisable par n'importe quel robot équipé de capteurs similaires
— Le robot remplit les objectifs fixés : lorsqu'il ne perçoit pas d'humains, il poursuit sa patrouille. Dès qu'un humain est vu ou entendu, le robot cherche à le localiser dans son environnement et à se positionner par rapport à lui.

Ce système pourrait toutefois être amélioré par différents aspects : actuellement, le robot ne garde pas en mémoire les évènements passés. Les évènements sont traités selon leur priorité et supprimés de la mémoire dès qu'un nouvel élément de priorité supérieure a lieu. Le robot ne peut donc pas revenir vers un événement précédemment perçu. Par ailleurs, le potentiel de fusion de données n'est pas exploité pleinement : il est intéressant d'arriver à tirer du sens de l'information perçue dans le temps. Il est donc souhaitable d'utiliser un formalisme de fusion pour les grilles, incluant en permanence toutes les observations capteurs. C'est ce que l'on se propose de faire dans le prochain chapitre.

Chapitre 5

Modélisation crédibiliste de l'environnement pour la navigation

Sommaire

5.1 Introduction 125
5.2 État de l'art sur le SLAM évidentiel 126
5.2.1 Grilles évidentielles 127
5.2.2 SLAM évidentiel à maximum de vraisemblance 128
5.2.3 SLAM évidentiel complet 131
5.2.4 Synthèse sur le SLAM évidentiel 134
5.3 Cartographie évidentielle incluant l'être humain 134
5.3.1 Présentation de l'architecture 134
5.3.2 Modèles capteurs inverses et construction des grilles évidentielles liées aux capteurs 136
5.3.3 Création de la grille évidentielle de perception à l'instant t 142
5.3.4 Fusion temporelle 143
5.4 Résultats 149
5.4.1 Résultats en simulation 149
5.4.2 Résultats sur le robot réel 158
5.5 Conclusion 169

5.1 Introduction

Dans le chapitre 4, un système de navigation automatique ayant pour but de détecter et de se positionner par rapport à des humains a été présenté, et implémenté sur un robot. Il repose sur la construction de grilles simples, prenant en compte les informations fournies par différents capteurs. Les informations sont agrégées en une unique grille, et seule l'information considérée comme étant prioritaire (un humain détecté dans la vidéo représentant la priorité maximale) est prise en compte lors de la décision à chaque instant.

Dans ce chapitre, la mission du robot reste la même : explorer son environnement à la recherche d'humains et se positionner par rapport à eux, dans le but ultérieur d'interagir avec eux. Toutefois on souhaite ici améliorer la manière dont la grille de perception multimodale est construite au cours du temps. Trois axes d'amélioration sont explorés ici :

— La prise en compte de l'incertitude liée aux mesures capteurs et à la détection vidéo, via l'utilisation du cadre évidentiel. Les fonctions de croyance disposent d'opérateurs de fusion plus raffinés qu'une simple agrégation des données au sein d'une carte, et une intégration temporelle permet de garder en mémoire l'information perçue pendant le temps désiré.
— L'indépendance complète entre la perception et la décision par le robot : l'intégralité des informations perçues doit être prise en compte à chaque instant, avant de prendre une décision sur le prochain mouvement du robot.
— La prise en compte de la dynamique de la scène : un robot compagnon ne doit pas représenter une gêne pour les personnes de son environnement, et ne se comportera donc pas de la même manière avec une personne mobile qu'avec une personne arrêtée. Il est donc intéressant de modéliser cet aspect au sein du système.

Pour créer une grille prenant en compte les caractéristiques ci-dessus, on propose un système de cartographie reposant sur une fusion évidentielle de grilles extraites des différentes modalités : l'information de profondeur et la vidéo extraites du capteur RGB-D, le flux audio extrait des microphones, et le scan laser extrait du télémètre. Cette fusion permet ainsi au robot d'obtenir une représentation compacte de son environnement, sous forme de grille métrique. La grille peut être utilisée pour prendre une décision de manière proche de ce qui a été présenté dans le chapitre 4. Ce travail a fait l'objet d'une publication [Lab+16].

L'utilisation des fonctions de croyance pour la cartographie ayant déjà été explorée dans la littérature, le présent chapitre est divisé comme suit : la section 5.2 présente un état de l'art concis des travaux existants sur le SLAM évidentiel, puis la section 5.3 détaille la méthode de cartographie proposée. Finalement, les résultats du système sont montrés dans la section 5.4.

5.2 État de l'art sur le SLAM évidentiel

La cartographie, via le problème du SLAM (pour Simultaneous Localization And Mapping), a été largement abordée du point de vue probabiliste, comme il a été montré dans la section 4.2. Pour rappel, le problème du SLAM revient à estimer à chaque instant d'observation la distribution de probabilités suivante :

$$p\left(m, x_t | z_{1:t}, u_{1:t}\right) \tag{5.1}$$

où m est une grille d'occupation où chaque cellule peut se trouver dans deux états réels différents : occupée ou vide ; x_t est la position du robot dans cette carte à l'instant t, $z_{1:t}$ est un ensemble d'observations de l'environnement effectuées par le robot au cours du temps, et $u_{1:t}$ est un ensemble de commandes données au robot au cours du temps pour le faire bouger. Ainsi, on estime à chaque instant l'état réel de chaque cellule sur l'ensemble d'états $\{O, F\}$, où O est l'état "occupé" (occupied), et F est l'état "libre" (free). Une distribution de probabilités sur cet espace d'états peut donc être résumée par une valeur d'occupation comprise entre 0 et 1 représentant la probabilité estimée sur l'état O (le degré de "liberté" d'une cellule étant le complémentaire à 1). Cependant, ce type de représentation présente des limites. En effet,

il est nécessaire de garder l'information perçue interprétable par un robot. Un cas simple, qui pourrait poser problème pour une décision, est le cas où le robot ignore complètement l'état d'une cellule (par exemple à cause d'une inconsistance dans les mesures, ou parce que la cellule considérée se trouve en dehors du champ de perception du robot), la distribution de probabilités correspondante est une distribution uniforme ($P(O) = 0.5$ et $P(F) = 0.5$). Seulement, cette information est difficilement interprétable par un robot. Une solution pour obtenir une carte sur laquelle le robot prend des décisions est un seuillage : au-delà d'un certain seuil d'occupation ($P(O) > \alpha$ ARBitraire, $\alpha \in [0, 1]$), on considère que la cellule est occupée. Cependant, l'ignorance n'est pas représentée explicitement dans ce genre de situation. Or elle peut représenter un intérêt pour un robot autonome : par exemple, ne pas être en mesure de dire ce que contient une partie de la carte peut encourager un robot à s'approcher et explorer plus en détail cette partie de son environnement. Pour augmenter le pouvoir de représentation des grilles d'occupation, la notion de grille évidentielle a été introduite dans la littérature.

5.2.1 Grilles évidentielles

Les grilles évidentielles ont été introduites par [PNDW96] pour construire des grilles d'occupation avec des capteurs à ultrasons. Une grille évidentielle est définie comme une grille où chaque cellule contient une fonction de croyance. Ainsi pour un espace de discernement défini comme $\Omega = \{O, F\}$, chaque cellule contient 4 valeurs correspondant aux différentes hypothèses de l'ensemble des parties de l'espace de discernement, comme montré par la figure 5.1.

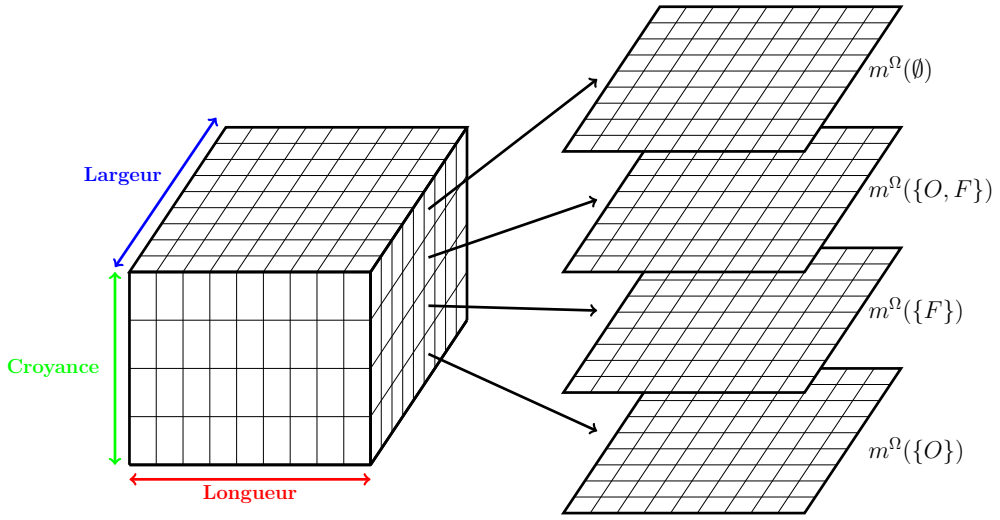


FIGURE 5.1 – Exemple d'une grille évidentielle sur l'espace de discernement $\Omega = \{O, F\}$

L'ignorance et le conflit entre deux mesures successives peuvent donc être directement modélisés, à travers la croyance attribuée aux hypothèses $\emptyset$ (le conflit) et $\{O, F\}$, et utilisés par le robot comme des informations pertinentes. Cependant cette amélioration nécessite de maintenir une fonction de croyance sur un nombre plus grand d'hypothèses (2^N, où N est le nombre d'hypothèses dans l'espace de discernement), et implique en conséquence un coût de

calcul généralement plus élevé.

La construction des grilles évidentielles a fait l'objet de plusieurs travaux dans la littérature, et est souvent désigné comme du "SLAM évidentiel". Comme pour le cas probabiliste, on trouve des exemples de travaux pour du SLAM évidentiel complet, et d'autres pour du SLAM évidentiel "à maximum de vraisemblance", plus adapté à une construction de carte en ligne. Les deux approches sont présentées ici.

5.2.2 SLAM évidentiel à maximum de vraisemblance

L'approche à maximum de vraisemblance du SLAM dans le cadre probabiliste a été décrite dans la section 4.2. Elle se divise en deux étapes successives (figure 5.2) : tout d'abord trouver l'état le plus vraisemblable du robot dans la grille d'occupation (sa position). Ceci est fait en calculant la valeur suivante :

$$\widehat{x}_t = \underset{x_t}{\operatorname{argmax}}\, p\left(x_t | z_t, u_t, \widehat{x}_{t-1}, \widehat{m}_{t-1}\right) = \underset{x_t}{\operatorname{argmax}}\left\{p(z_t | x_t, \widehat{m}_{t-1}) p(x_t | u_t, \widehat{x}_{t-1})\right\} \tag{5.2}$$

Cette opération est appelée "scan-matching", car elle calcule la valeur la plus vraisemblable, prenant en compte la grille à l'instant t-1 et les observations à l'instant t. Ensuite, la grille est mise à jour avec l'information perçue à l'instant t :

$$m_t = \widehat{m}_{t-1} \cup \{< \widehat{x}_t, z_t >\} \tag{5.3}$$

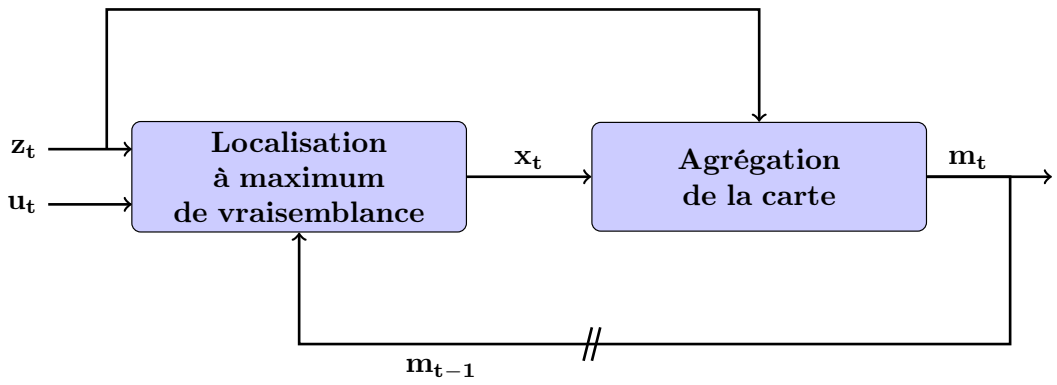


FIGURE 5.2 – Processus général de SLAM à maximum de vraisemblance

Pour mettre à jour la grille, il est nécessaire de la construire à partir des données capteurs. Dans le cadre probabiliste, cela revient à connaître la distribution de probabilités $\mathrm{p}(m_t | x_t, z_t)$. C'est le **modèle capteur inverse**. L'avantage du SLAM à maximum de vraisemblance tient principalement dans le fait qu'il est peu coûteux en calcul, car il ne maintient qu'une unique

carte à chaque instant d'observation, en ne prenant en compte que la position la plus vraisemblable du robot, plutôt qu'une trajectoire complète possible.

C'est cette approche qui est généralement employée dans les travaux de SLAM évidentiel : estimation de la position la plus vraisemblable, puis mise à jour de la carte. En effet, l'estimation conjointe de la carte et de la trajectoire complète du robot a été à ce jour peu explorée, et sera décrite dans la sous-section suivante.

[Tre+14] propose par exemple un travail de SLAM évidentiel à maximum de vraisemblance, avec un espace de discernement classique : $\Omega = \{F, O\}$, et se concentre principalement sur l'opération de scan-matching. Pour calculer la position la plus vraisemblable du robot, deux étapes ont lieu : tout d'abord, des positions candidates sont tirées à l'aide du modèle dynamique du robot, auquel une incertitude est ajoutée pour diversifier les candidats (la méthode de proposition de candidats a été améliorée en un filtrage particulaire dans [Tre+15]). Pour chacun de ces candidats, un score est calculé de la manière suivante :

$$Score = \sum_{\forall cells} Op\left(\widehat{m}^{\Omega}_{i,j,t-1}, \widetilde{m}^{\Omega}_{i,j,t}\right) \tag{5.4}$$

où :

— Op est un opérateur de fusion de fonctions de croyance (conjonctif, disjonctif, combinaison de Dempster, etc...)
— $\widehat{m}^{\Omega}_{i,j,t-1}$ représente la cellule d'index (i, j) dans la grille évidentielle estimée à l'instant $t-1$ sur l'espace de discernement Ω.
— $\widetilde{m}^{\Omega}_{i,j,t}$ représente la cellule d'index (i, j) dans la grille "mesurée" à l'instant t pour la position candidate.

La grille $\widetilde{m}$ représente le modèle capteur inverse, soit la grille construite à partir des données capteurs. Pour un laser elle est généralement construite en traçant chaque rayon du laser (opération de "ray-tracing"). Toute cellule traversée par un rayon est considérée comme libre et reçoit une fonction de croyance adéquate. Les cellules correspondant à un point d'impact du laser reçoivent une fonction de croyance avec une croyance forte sur O [1]. Les cellules placées en dehors du champ de perception reçoivent une fonction de croyance vide ($m(\Omega) = 1$). La figure 5.3 illustre un exemple d'un tel modèle capteur inverse. Dans ce cas, les cordonnées des cellules sont polaires (une conversion en coordonnées cartésiennes est faite ultérieurement).

Ainsi le modèle capteur inverse proposé par [Tre+14] est donné mathématiquement par :

$$\begin{cases} \widetilde{m}_{r,\theta,t}(A) &= \lambda \\ \widetilde{m}_{r,\theta,t}(\Omega) &= 1-\lambda \end{cases} \quad \text{avec} \quad \begin{cases} A = O & \text{si cellule impactée} \\ A = F & \text{si cellule traversée} \\ \lambda = 0 & \text{si cellule non-perçue} \end{cases} \tag{5.5}$$

1. A nouveau, dans ce chapitre, les sous-ensembles singletons seront notés sans accolades pour faciliter la lecture

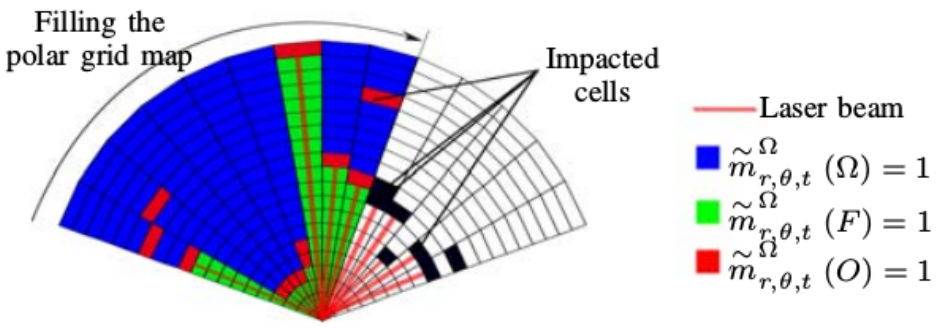


FIGURE 5.3 – Exemple de remplissage d'une grille polaire avec un scan laser, tirée de [Tre+14]

où $\widetilde{m}_{r,\theta,t}$ est la fonction de croyance attribuée à la cellule de coordonnées (r, θ) dans la grille polaire, et λ est une valeur pré-déterminée correspondant à la confiance faite au capteur. Si la confiance est forte, λ sera proche de 1, et inversement si le capteur est peu fiable, λ sera proche de 0.

Ainsi [Tre+14] calcule un score pour chaque position du robot proposée et la position obtenant le score le plus élevé est considérée comme la plus vraisemblable. Plusieurs opérateurs Op de scan-matching sont proposés, correspondant à des opérateurs de fusion crédibiliste : fusion disjonctive, disjonctive normalisée, etc...

La mise à jour de la grille est faite à l'aide d'une fusion de Dempster, entre la grille estimée à l'instant $t-1$ et la grille mesurée à l'instant t :

$$\widehat{m}^{\Omega}_{i,j,t}(A) = \begin{cases} \frac{\sum_{B \cap C = A} \widehat{m}^{\Omega}_{i,j,t-1}(B) \cdot \widetilde{m}^{\Omega}_{i,j,t}(C)}{1 - \widehat{m}^{\Omega}_{i,j,t}(\emptyset)} & \text{si A} \neq \emptyset \\ 0 & \text{sinon} \end{cases} \tag{5.6}$$

Cette approche pose des bases pour un SLAM évidentiel à maximum de vraisemblance mais présente plusieurs inconvénients : le coût calculatoire de la partie de scan-matching peut être rapidement très élevé, au fur et à mesure que la grille grandit. En effet, elle nécessite de fusionner les grilles cellule par cellule, autant de fois qu'il y a de positions candidates proposées par le modèle dynamique. De plus, la mise à jour de la grille gère le conflit de la manière la plus simple existante, sans en tirer du sens. Cette approche présente toutefois des résultats en précision de positionnement du robot supérieurs à l'approche probabiliste.

D'autres travaux se sont intéressés plus en détail à la mise à jour de la grille entre l'instant $t-1$ et l'instant t, notamment dans le cadre de travaux sur la construction de grille par des véhicules autonomes. [MCB11] propose par exemple d'exploiter le conflit d'informations capteur entre la grille précédemment construite et la grille mesurée pour détecter les objets mobiles. Le véhicule dispose ici d'un unique laser en guise de capteur extéroceptif. La position

du véhicule est donnée par un système de positionnement expert : on considère qu'à chaque fois la position la plus vraisemblable est fournie au système. La notion d'objets mobiles n'est pas directement représentée dans l'espace de discernement utilisé, qui reste $\Omega = \{O, F\}$.

Ces travaux ont été poursuivis par [Kur+14 ; Kur+15], toujours dans le cadre d'un travail sur les véhicules autonomes en milieu urbain. Dans ce travail, il a été proposé d'agrandir le cadre de discernement pour augmenter la quantité d'informations que peut contenir une grille évidentielle (figure 5.4). Pour ce faire, un capteur laser est utilisé, donnant l'information sur l'espace de discernement classique : $\Omega_1 = \{O, F\}$, mais les auteurs utilisent aussi l'information provenant de cartes externes de la zone urbaine à cartographier, fournies par l'Institut National de l'Information Géographique. Cela permet d'étendre l'espace de discernement et d'inclure de nouveaux types d'informations : un nouvel espace de discernement est amené par ces cartes, $\Omega_2 = \{B, R, T\}$, où B représente les immeubles (buildings), R les routes (roads), et T les espaces intermédiaires (trottoirs, espaces entre immeubles, etc...). Les informations fournies par le laser et les informations connues a priori sont fusionnées dans le temps dans un espace de discernement final comprenant les objets mobiles, les différents types d'espaces urbains, et plusieurs types d'infrastructures.

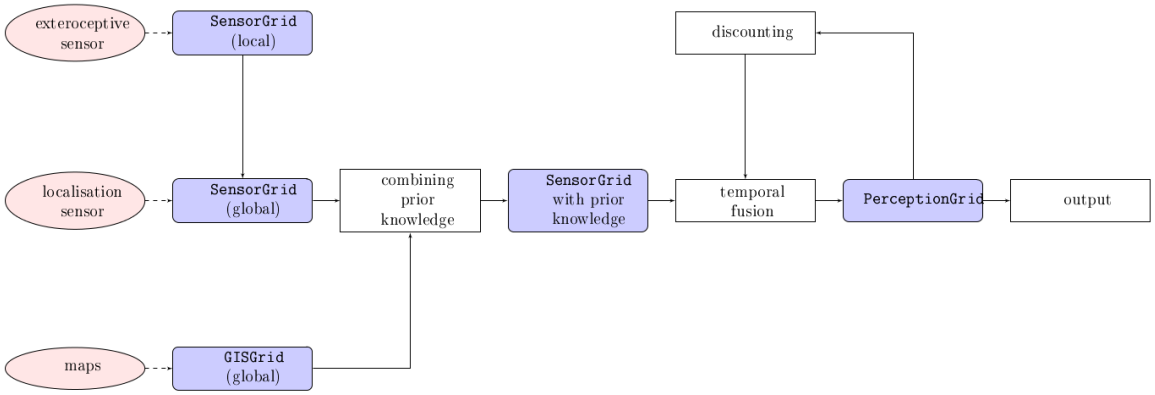


FIGURE 5.4 – Architecture proposée par [Kur+15]

Cette approche comporte des atouts non-négligeables, dont une réelle augmentation du pouvoir de représentation des cartes, à l'aide d'information extérieure connue a priori. Un véritable cadre de fusion de l'information est mis en place, et les objets mobiles sont ajoutés directement à l'espace de discernement. L'approche proposée peut être étendue à plusieurs capteurs extéroceptifs, mais n'a été testée qu'avec un laser. Ici encore, la position du véhicule est considérée comme connue précisément à chaque instant.

5.2.3 SLAM évidentiel complet

Le SLAM complet, comme décrit dans la section 4.2, consiste à estimer conjointement la trajectoire complète et la carte de l'environnement du robot, soit du point de vue probabiliste

la distribution :

$$p(x_{1:t}, m|z_{1:t}, u_{1:t}) \tag{5.7}$$

Comme dit précédemment, il existe peu d'exemples de SLAM complet évidentiel, à l'exception de [RC13], puis dans une version plus détaillée [CRK16], qui proposent une adaptation du filtre particulaire Rao-Blackwellisé généralement utilisé pour résoudre le problème de l'estimation conjointe. Dans ce travail, l'adaptation du filtre particulaire au cadre évidentiel proposée par [Ren14 ; Rei11] est utilisée pour résoudre le problème du SLAM sur l'espace de discernement $\Omega = \{O, F\}$. La décomposition du filtre particulaire est donnée comme suit :

$$m(x_{1:t}, Y|z_{1:t}, u_{1:t}) = p(x_{1:t}|z_{1:t}, u_{1:t})m(Y|x_{1:t}, z_{1:t}) \tag{5.8}$$

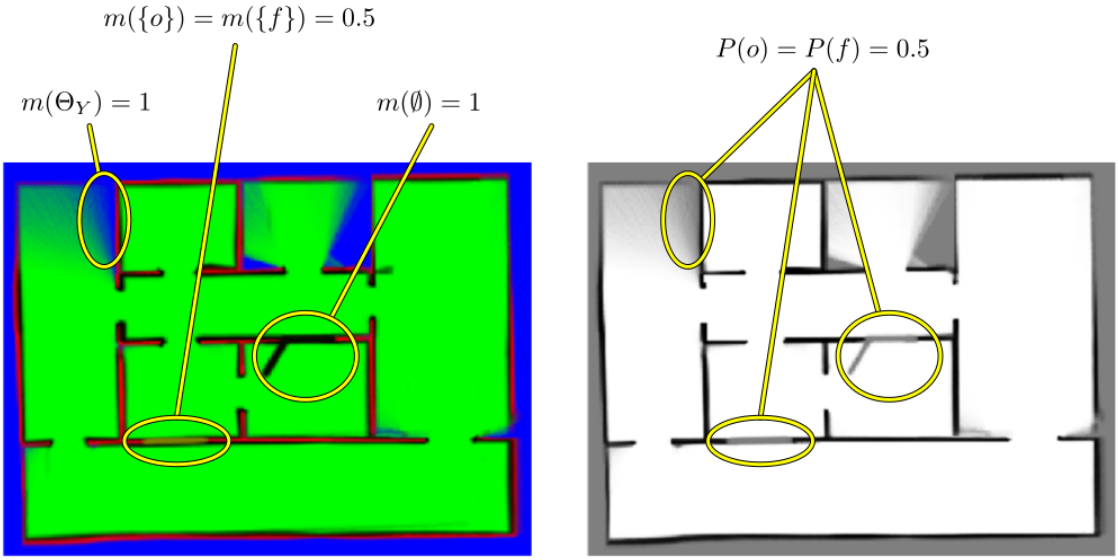
où :
— Y représente la grille évidentielle estimée,
— $m(x_{1:t}, Y|z_{1:t}, u_{1:t})$ représente une distribution de masses de croyance, et non la grille. C'est donc la distribution de masses sur les positions $x_{1:t}$ du robot au cours du temps et la grille Y, connaissant les observations successives $z_{1:t}$, et les commandes successives $u_{1:t}$ données au robot.

La forme compacte de l'estimée conjointe est très proche de l'approche probabiliste (équation 4.5), avec une séparation de l'équation en deux parties, tout comme le SLAM à maximum de vraisemblance. Tout d'abord un ensemble de trajectoires possibles du robot est estimé (calcul de $p(x_{1:t}|z_{1:t}, u_{1:t})$), puis le modèle capteur inverse est utilisé pour mettre à jour la carte pour chacune des trajectoires possibles calculées (calcul de $m(Y|x_{1:t}, z_{1:t})$). Tout comme le cas du SLAM complet probabiliste, le SLAM complet évidentiel consiste à maintenir en permanence un nombre de grilles égal aux nombre de positions candidates tirées. Le SLAM complet évidentiel revient donc à maintenir en permanence plusieurs cartes de SLAM à maximum de vraisemblance.

L'originalité de [CRK16] repose dans la formulation du modèle capteur (utilisé pour la localisation du robot), ainsi que du modèle capteur inverse (utilisé pour la mise à jour de la carte, de la même manière que dans le SLAM à maximum de vraisemblance). Le modèle capteur est calculé mathématiquement en prenant en compte la probabilité de défaut du capteur (probabilité d'avoir une mesure sans aucun sens), et le calcul dans le cadre évidentiel donne naturellement des résultats similaires au cadre probabiliste.

Cette méthode est adaptée à la construction d'une grille contenant les objets statiques de l'environnement du robot, dans laquelle il peut se localiser. L'avantage par rapport au cadre probabiliste est l'augmentation du pouvoir de représentation. La figure 5.5 illustre ceci : sur la droite, on voit que tous les problèmes de conflit capteur sont résolus par le cas $P(o) = P(f) = 0.5$. Dans le cadre évidentiel, des distinctions apparaissent : lorsque le robot n'a pas eu l'occasion de percevoir une partie de la carte, l'ignorance est mise à 1, tandis que si

l'environnement a changé entre plusieurs mesures successives (cas d'une porte), les cellules correspondantes comportent une distribution de masses où le conflit l'emporte.



(a) evidential grid map (b) probabilistic grid map

FIGURE 5.5 – Comparaison entre la grille évidentielle proposée par [CRK16] (a) et la méthode de SLAM probabiliste de [GSB07] (b). Figure tirée de [CRK16]. Sur la grille (a), le vert représente la croyance sur l'hypothèse F (espace libre navigable), le bleu la croyance sur Θ_Y (l'espace de discernement, soit l'ignorance), et le rouge la croyance sur O (objet). Sur la grille (b), le blanc représente l'espace libre navigable F, et le noir la présence d'objet O.

Une autre originalité de [CRK16] est l'élaboration d'un algorithme de planification de trajectoire dans une grille évidentielle. Une adaptation d'un algorithme de planification classique est proposée : une version évidentielle des processus de décision markovien. L'idée de la planification de trajectoire à l'aide de processus de décision markovien sur des grilles probabilistes est de construire une fonction de "récompense", représentant le coût total associé à un chemin parcouru sur la grille. La plupart du temps, la récompense associée à une trajectoire dépend de deux choses : la distance totale parcourue (qu'on cherche à minimiser pour trouver le chemin le plus court au sein de la carte), et le nombre d'obstacles traversés (qu'on cherche à minimiser pour éviter des collisions au robot). Ainsi, pour planifier un chemin d'un point à un autre de la carte, on cherche la trajectoire maximisant la récompense totale. [CRK16] propose une adaptation de ce processus au cadre évidentiel, en prenant en compte le fait que chaque cellule contient une distribution de masses, et que le coût de passage par une cellule doit prendre en compte les valeurs $m(\emptyset)$ représentant le conflit entre des mesures capteurs dans le temps et $m(\{O, F\})$ représentant l'ignorance du contenu d'une celulle.

Cet algorithme de planification est adaptée à une planificiation globale sur une grille évidentielle déjà construite, et ne prend pas en compte des mesure intermédiaires.

5.2.4 Synthèse sur le SLAM évidentiel

Les deux types d'approche de SLAM ont donc été explorés dans le cadre évidentiel. Toutefois, le SLAM complet semble moins adapté à l'approche évidentielle que le SLAM à maximum de vraisemblance : pour construire une carte de l'environnement statique, qui servira la plupart du temps pour la planification de trajectoire globale, une grille probabiliste apparaît suffisante : en effet, une fois la carte statique construite, le robot ne peut pas agir sur les zones perçues contenant du doute, ou du conflit. Dans le cas de [CRK16], le niveau d'interprétation supplémentaire n'est utilisé que pour ajuster le coût d'une trajectoire, alors qu'on pourrait imaginer le mettre à profit pour éviter les endroits ayant provoqué du conflit ou du doute pour le robot.

L'information supplémentaire apportée par le cadre évidentiel pourrait être plus facilement exploitée dans le cadre du SLAM évidentiel à maximum de vraisemblance, car le robot peut agir directement sur la construction de sa carte en allant explorer les informations jugées utiles. L'exemple apporté par [Kur+15] illustre ce problème, car il utilise des données connues a priori pour ajouter du sens aux données capteurs.

Toutefois dans cet exemple, la multimodalité n'est pas utilisée. Dans notre cas, on dispose de capteurs de natures différentes dont on souhaite inclure les informations directement dans une carte. Le robot a une mission précise : il doit être capable de détecter les humains dans son environnement et de se positionner par rapport à eux. Il est donc important que sa représentation du monde inclue directement les humains détectés, avec l'incertitude liée à la détection. C'est un tel système qu'on se propose de présenter dans la section 5.3. Il repose sur une méthode de SLAM à maximum de vraisemblance.

5.3 Cartographie évidentielle incluant l'être humain

Nous présentons un système de cartographie multimodale adaptée à la détection d'humains par un robot mobile à l'aide d'un laser, d'un capteur RGB-D et d'une paire de microphones. Elle est divisée en plusieurs parties : la section 5.3.1 présente l'architecture générale du système proposé, ainsi que les hypothèses qui sont faites par rapport aux conditions dans lesquelles se trouve le robot. Les sections 5.3.2, 5.3.3 et 5.3.4 présentent les étapes majeures de l'architecture, et finalement une démonstration du système, en simulation puis sur un robot réel, est visible dans la section 5.4.

5.3.1 Présentation de l'architecture

On se place dans un contexte similaire à celui du chapitre 4 : on considère que le robot a déjà à disposition une grille d'occupation probabiliste, construite à l'aide d'un algorithme de SLAM complet lors d'un passage précédent dans son espace de travail. Cette grille, gardée en

mémoire est utilisée uniquement pour la localisation. Ainsi à chaque instant t, on a accès à la position x_t la plus vraisemblable du robot dans la grille. Ceci est l'équivalent de l'étape de "scan-matching" du SLAM à maximum de vraisemblance. Le robot, connaissant sa position doit construire une grille évidentielle contenant les informations de présence d'obstacles ainsi que d'êtres humains dans son environnement, et la maintenir dans le temps.

L'architecture présentée ici (figure 5.6), se concentre sur la mise à jour de la grille au cours du temps. On suppose qu'une détection d'êtres humains est effectuée sur chaque trame vidéo. De même, la localisation de la source sonore a lieu à chaque instant d'observation.

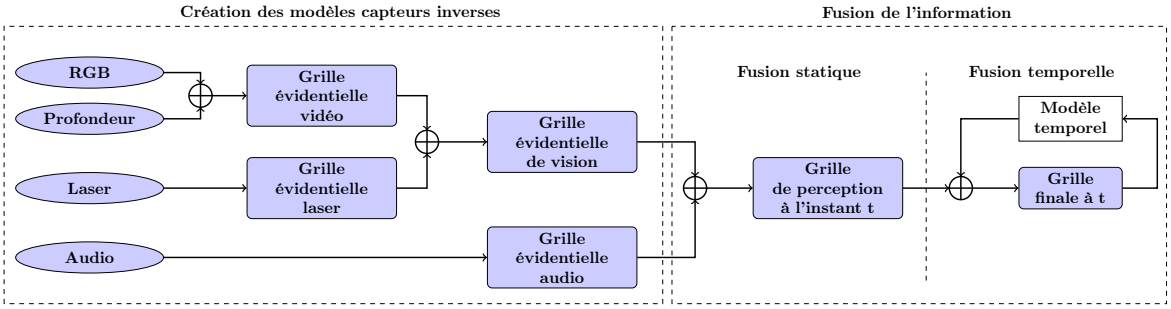


FIGURE 5.6 – Architecture proposée pour la cartographie

On peut diviser le système proposé en deux grandes sous-parties, similaires à [Kur+15] : tout d'abord une grille est construite pour chaque capteur à chaque instant d'observation, représentant pour chacun l'information apportée. L'information de type visuel (laser et vidéo) est agrégée en une grille de vision. Puis vient la phase de fusion de l'information perçue sur l'ensemble des capteurs.

La grille de perception obtenue est intégrée dans un schéma temporel mettant à jour la grille en construction. Cette grille peut par la suite être utilisée pour décider du prochain mouvement du robot et planifier sa trajectoire.

L'objectif étant de construire une grille de l'environnement contenant de l'humain, une cellule peut donc se trouver dans trois états différents. Chaque cellule de nos grilles évidentielles peut donc recevoir une distribution de masses sur l'espace de discernement $\Omega = \{H, O, F\}$, où :

— H : présence d'une être humain au sein de cette cellule,
— O : présence d'un objet au sein de cette cellule (part de l'environnement statique, ou mobile non-humain),
— F : la cellule est vide, et contient de l'espace libre navigable par le robot.

5.3.2 Modèles capteurs inverses et construction des grilles évidentielles liées aux capteurs

Une grille est construite à chaque instant d'observation, pour chacun des capteurs. Pour cela, on fait appel à des modèles capteurs inverses, définis pour chaque capteur. Chaque cellule des grilles estimées à l'aide des modèles capteurs inverses doit donc contenir une distribution de masses représentant au mieux les informations capteur perçues : c'est la masse de $m(Y, x_t|z_t)$ proposée par [CRK16], dans la section 5.2.

Pour illustrer les modèles capteurs inverses proposés ici et la construction de grilles, on se propose d'utiliser le même exemple jouet que dans le chapitre 4, où un robot se trouve dans une grille en présence d'un être humain et d'obstacles statiques. Cet exemple est illustré par la figure 5.7 : il contient les trois types d'informations possibles, les cellules blanches représentant l'espace libre (F), les cellules rouges l'humain (H), et les cellules noires les objets de l'environnement (O).

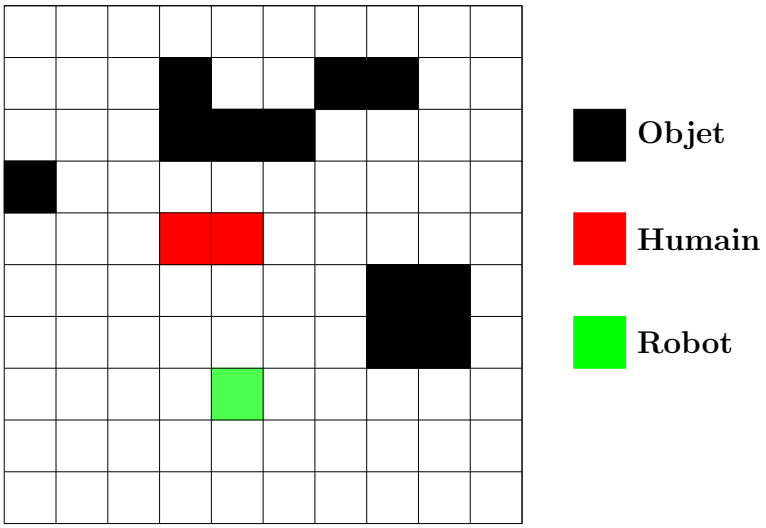


FIGURE 5.7 – Exemple jouet d'une grille d'occupation, représentant l'état réel d'une scène

5.3.2.1 Construction de la grille évidentielle vidéo

Cette grille est construite à partir du capteur RGB-D. Or comme on l'a vu dans le chapitre précédent (section 4.3.2.1), il est possible grâce au capteur de profondeur de replacer une observation dans le système de coordonnées capteurs. Connaissant la position du capteur sur le robot, l'observation peut par la suite être replacée dans la grille. Si on ne prend en compte que les détections d'humains, au sein d'une cellule, le capteur RGB-D peut donc donner deux types de renseignements :

— Possible présence d'un humain dans la cellule, si une détection est replacée dans la cellule (hypothèse H),

— Ignorance à propos de l'état de la cellule, si aucune détection n'a été faite dans la cellule (hypothèse $\Omega = \{H, O, F\}$).

En transformant l'information de profondeur en données télémétriques de la même façon que décrite dans le chapitre précédent, il est aussi possible d'obtenir un pseudo-scan laser supplémentaire renseignant sur la possible présence d'objets de l'environnement. Cette information pourrait être utile dans plusieurs cas : par exemple, si le capteur RGB-D et le laser ne se trouvent pas à la même hauteur, ils peuvent apporter des informations complémentaires et aider le robot à percevoir des obstacles invisibles pour un des deux capteurs (le robot étant de petite taille, ça peut être le cas pour l'empêcher de percuter une chaise trop haute que le laser ne détecterait pas). Cependant, le replacement spatial des observations n'étant pas consistant dans le temps, comme on l'a vu au chapitre précédent, on préfère se concentrer sur les observations spécifiques au capteur RGB-D (soit la détection d'êtres humains), pour ne pas apporter de bruit à la grille en ce qui concerne les objets de l'environnement.

Pour ce qui est du modèle capteur inverse associé, une fois le replacement fait, on choisit une distribution de masses classique. Pour une cellule donnée :

$$\left\{ \begin{array}{rcl} m^{\Omega}_{vidéo}(H) & = & \lambda \\ m^{\Omega}_{vidéo}(\Omega) & = & 1-\lambda \end{array} \right. \quad \text{avec} \quad \left\{ \begin{array}{cc} 0 < \lambda < 1 & \text{si détection} \\ \lambda = 0 & \text{sinon} \end{array} \right. \tag{5.9}$$

Le paramètre λ représente la confiance faite à l'algorithme de détection employé, et plus précisément dépend de la quantité moyenne de faux positifs. Si, lorsqu'une détection a lieu, la confiance dans le détecteur est grande, alors λ sera proche de 1. Dans le cas contraire, plus la confiance sera faible, plus λ sera proche de 0.

On note ici que l'imprécision spatiale, c'est à dire la qualité de replacement de l'observation vidéo dans la grille, n'est pas représentée : on s'intéresse à la construction de la grille cellule par cellule, et une observation ne peut généralement être associée qu'à une unique cellule. Cependant il est possible d'associer une observation à plusieurs cellules pour exprimer l'imprécision du capteur, c'est à dire de créer une gaussienne discrétisée spatialement, dont la croyance sur H décroit en s'éloignant de la cellule où l'observation a été faite.

La grille évidentielle vidéo de l'exemple jouet est représentée par la figure 5.8.

5.3.2.2 Construction de la grille évidentielle laser

Cette grille est construite à partir des données du capteur laser. Le laser est le capteur qui a été le plus utilisé pour la cartographie et les modèles capteurs inverses associés sont classiques. Comme décrit dans la section 5.2, pour construire la grille, on utilise généralement une méthode dite de "lancer de rayon" ("ray-tracing" en anglais). Pour ce faire, chaque rayon laser est tracé dans la grille jusqu'à son point d'impact (s'il existe). La figure 5.9 représente le lancer de rayon sur l'exemple, pour les rayons les plus importants.

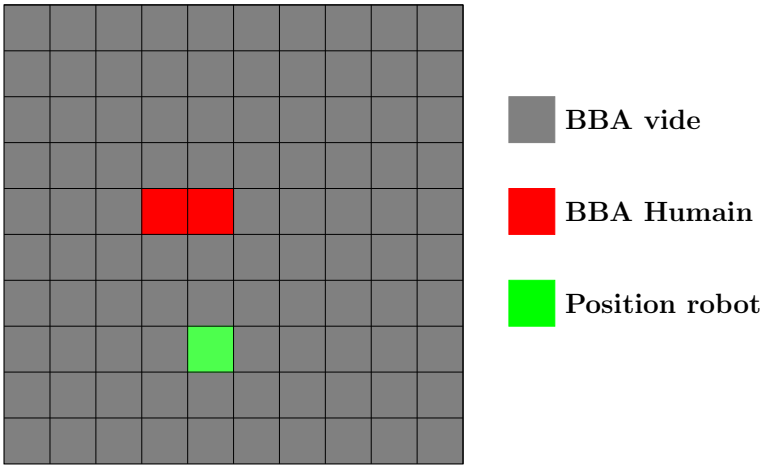


FIGURE 5.8 – Exemple jouet d'une grille évidentielle vidéo obtenue à l'aide du capteur RGB-D. Chaque cellule peut recevoir 2 types de BBA (distribution de masses de croyance) dépendant du fait qu'une détection a eu lieu ou non au sein de cette cellule.

On voit qu'il existe trois cas possibles pour une cellule :
- La cellule est traversée par un rayon, ce qui en fait une cellule libre pour le capteur (hypothèse F),
- La cellule est le lieu d'un impact de rayon avec un objet de l'environnement, ce qui en fait une cellule contenant un objet ou un humain (le laser étant incapable de distinguer entre les deux) (hypothèse $\{H, O\}$),
- La cellule ne fait pas partie du champ de perception du capteur, ce qui ne permet pas de qualifier son état (hypothèse $\Omega = \{H, O, F\}$).

Ces trois possibilités donnent trois distributions de masses possibles selon les cas :

BBA Objet ou Humain	**BBA Espace navigable**	**BBA vide**
$\begin{cases} m^{\Omega}_{laser}(\{H,O\}) = \lambda_1 \\ m^{\Omega}_{laser}(\Omega) = 1-\lambda_1 \end{cases}$	$\begin{cases} m^{\Omega}_{laser}(F) = \lambda_2 \\ m^{\Omega}_{laser}(\Omega) = 1-\lambda_2 \end{cases}$	$m^{\Omega}_{laser}(\Omega) = 1$

(5.10)

Les paramètres λ_1 et λ_2 ne dépendent plus de la performance d'algorithmes mais de la confiance capteur. Si la confiance dans le capteur laser est forte, λ_1 et λ_2 auront une valeur proche de 1, et inversement si la confiance est faible leur valeur s'approchera de 0. Ces valeurs sont adaptables et peuvent varier en fonction de différents paramètres : par exemple, on peut adapter la confiance en fonction la distance au robot : plus la position de l'objet est proche des limites du capteur (et donc plus on s'éloigne du robot), moins le capteur peut être considéré comme fiable et les valeurs de λ_1 et λ_2 diminuent.

A partir du tracé de rayon de la figure 5.9, on peut donc construire la grille évidentielle laser, en attribuant à chaque cellule une BBA correspondant à son état perçu. Le résultat est montré sur la figure 5.10.

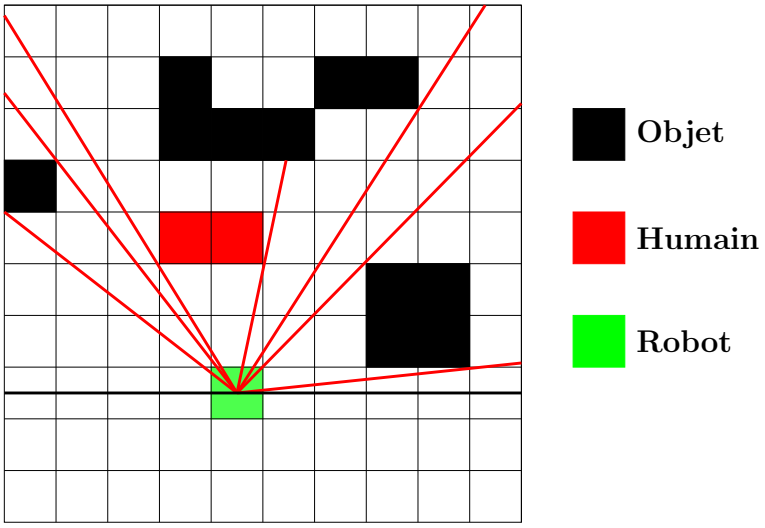


FIGURE 5.9 – Exemples de lancer de rayons : on considère ici que le capteur laser à un champ de vision de 180 degrés (tracé en noir).

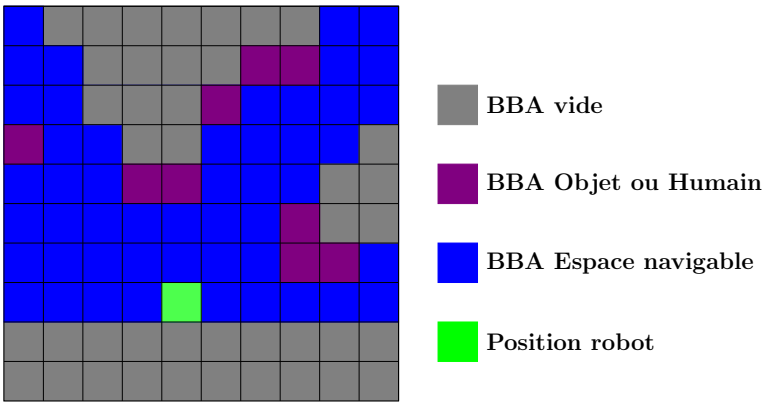


FIGURE 5.10 – Exemple jouet d'une grille évidentielle laser perçue par le capteur laser : chaque cellule peut recevoir trois types de BBA dépendant de son état perçu

5.3.2.3 Construction de la grille évidentielle de vision

La grille évidentielle de vision contient les informations liées à la vision dans son intégralité. Elle est donc le résultat de la fusion cellule par cellule de la grille évidentielle vidéo et de la grille évidentielle laser. Pour fusionner les deux grilles, on utilise une combinaison conjonctive, avec gestion du conflit en faveur du capteur RGB-D. Seul un cas de conflit peut avoir lieu (tableau 5.1).

Un conflit survient lorsqu'au sein d'une cellule un humain est détecté par de la vidéo (hypothèse H), sans qu'un objet soit détecté par le laser (hypothèse F). Cela peut correspondre à plusieurs cas de figure :

$m^{\Omega}_{vidéo}$ \ m^{Ω}_{laser}	$\{H, O\}$	F	Ω
H	H	$\emptyset$	H
Ω	$\{H, O\}$	F	Ω

TABLE 5.1 – Intersections possibles entre les différents éléments focaux d'une fonction de croyance m^{Ω}_{laser} associée à une cellule de la grille évidentielle laser et d'une fonction de croyance $m^{\Omega}_{vidéo}$ associée à une cellule de la grille évidentielle vidéo.

— Un humain a été détecté par le capteur RGB-D, alors qu'il n'était pas perceptible par le laser. Par exemple, si l'humain se trouve un peu en hauteur par rapport au capteur laser, aucune partie de son corps ne pourra être détectée par ce capteur.
— L'observation vidéo est mal localisée dans la grille, et correspond à une observation laser proche. Dans ce cas, ce serait prendre un risque que d'essayer d'associer l'observation vidéo avec l'observation laser la plus proche, car le capteur laser ne peut pas donner d'information sur la véracité d'une telle association. Il apparaît donc logique qu'en cas de conflit, on fasse confiance au capteur vidéo.

Si on choisit de faire confiance à la source laser, on prend non seulement le risque de ne pas détecter un être humain bien présent, mais en plus de rentrer en collision avec lui, ce qui est inacceptable. Ainsi pour fusionner les information laser et vidéo en une seule grille évidentielle de vision on utilise pour chaque cellule la règle de fusion conjonctive modifiée suivante :

$$m^{\Omega}_{vision}(S) = \left(m^{\Omega}_{vidéo} \textcircled{\cap} m^{\Omega}_{laser}\right)(S), \quad \forall S \subseteq 2^{\Omega}, S \neq H \tag{5.11}$$
$$m^{\Omega}_{vision}(H) = \left(m^{\Omega}_{vidéo} \textcircled{\cap} m^{\Omega}_{laser}\right)(H) + (m^{\Omega}_{a} \textcircled{\cap} m^{\Omega}_{v})(\emptyset) \tag{5.12}$$
$$m^{\Omega}_{vision}(\emptyset) = 0 \tag{5.13}$$

"Faire confiance au capteur RGB-D" en cas de conflit revient à transférer le conflit provoqué par la fusion sur l'hypothèse de la grille évidentielle vidéo ayant provoqué ce conflit, soit H. On obtient ainsi une grille évidentielle de vision représentant l'information perçue par les capteurs RGB-D et laser.

Au terme de la création de la grille, il n'y a que quatre éléments focaux possibles au sein de chacune des distributions de masses de la grille évidentielle de vision : H, $\{H, O\}$, F et Ω.

5.3.2.4 Construction de la grille évidentielle audio

La grille évidentielle audio est construite à partir de l'information perçue par les microphones. La localisation de source sonore est faite à partir de la paire de microphones dont est équipé le robot. Comme la classification de source audio n'a pas été implémentée sur le robot, on considère pour le moment que toute source audio provient d'un humain.

Comme on l'a vu dans le chapitre 3, localiser précisément une source avec deux microphones, sans fusion dans le temps ou mouvement des microphones est un problème difficile. On peut toutefois obtenir un angle en azimut par rapport à la position du robot, bien que cet angle soit souvent imprécis. Connaître la distance à laquelle se trouve la source sonore par rapport au robot est impossible. Dans le chapitre précédent, on représentait l'information apportée par l'audio comme un cône sur la grille dans lequel peut se trouver la source sonore. La taille du cône permet de quantifier l'incertitude spatiale de la mesure. On se propose d'utiliser ici la même représentation de l'information audio.

Il existe donc deux cas possibles pour une cellule donnée :
— La cellule se trouve dans un cône audio de détection : elle est alors candidate pour contenir un être humain
— La cellule ne se trouve pas dans un cône audio de détection : les microphones ne peuvent alors pas nous renseigner sur son état (ignorance complète)
Ces deux possibilités donnent ainsi deux distributions de masses possibles :

BBA cône	BBA vide
$\begin{cases} m^{\Omega}_{audio}(H) = \lambda \\ m^{\Omega}_{audio}(\Omega) = 1-\lambda \end{cases}$	$\{ \; m^{\Omega}_{audio}(\Omega) = 1$

(5.14)

λ est un paramètre reflétant la confiance accordée à la localisation audio. Le cône audio représentant une portion plutôt large de l'espace du robot, on ne peut accorder à cette modalité la même confiance qu'aux modalités de vision. Toutefois l'audio peut être un bon premier guide pour explorer plus en détail une région, de la même manière que dans le chapitre précédent, on explorait le cône audio à la recherche d'un être humain.

Le cône audio et la grille évidentielle correspondante sur l'exemple jouet sont visibles sur la figure 5.11.

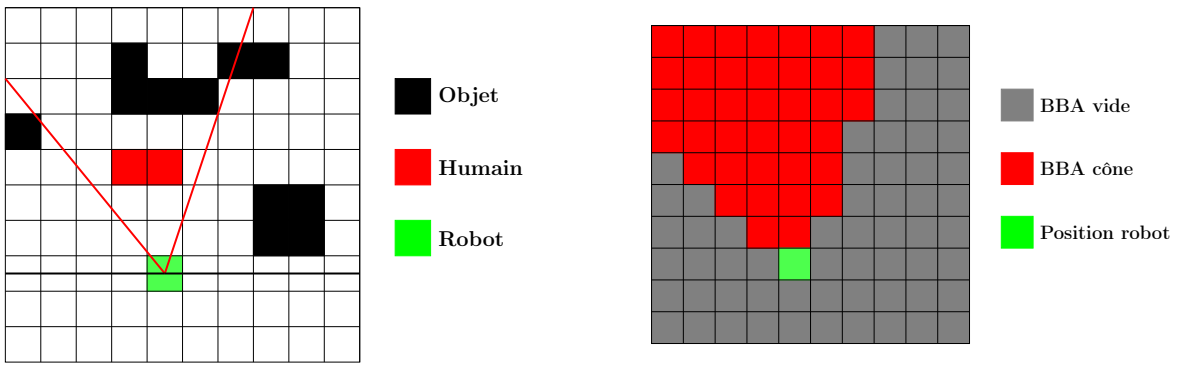


FIGURE 5.11 – Exemple jouet perçu par les microphones. Le cône audio est visible sur l'image de gauche en rouge. La grille évidentielle audio correspondante est visible sur l'image de droite.

5.3.2.5 Synthèse sur la construction des grilles évidentielles capteurs

Au terme de la construction de grilles évidentielles capteurs, on dispose donc de deux grilles : la grille évidentielle de vision, dont les cellules contiennent une distribution de masses de croyance m^{Ω}_{vision} correspondant à l'information perçue par les capteurs laser et RGB-D, et la grille évidentielle audio, dont les cellules contiennent une distribution de masses de croyance m^{Ω}_{audio} correspondant à l'information perçue par les microphones.

Ces deux grilles sont fusionnées pour obtenir une grille représentant l'information perçue sur l'ensemble des modalités.

5.3.3 Création de la grille évidentielle de perception à l'instant t

La grille de perception à l'instant t est le fruit de la fusion de la grille de vision et de la grille audio. L'étendue spatiale de l'information apportée par ces deux grilles peut être très différente. Pour fusionner l'information, il est nécessaire de prendre en compte deux éléments :

— On souhaite garder l'information interprétable par le robot : une cellule contenant une forte croyance sur l'élément $\{H, F\}$, ce qui signifierait "j'ai une croyance forte que cette cellule contient un humain ou de l'espace navigable" n'est pas acceptable. Elle représente une information contradictoire sur laquelle il est impossible de prendre une décision.
— On souhaite donner une forte priorité à la vision, car elle est spatialement plus précise que l'audition, ainsi que plus informative : elle permet de pointer directement dans l'environnement du robot un être humain détecté ou un obstacle perçu.

En prenant en compte ces deux éléments, c'est la fusion conjonctive qui encore une fois semble la plus adaptée : les éléments focaux des deux grilles à fusionner étant H et Ω d'un côté et H, $\{H, O\}$, F et Ω de l'autre, aucune intersection de ces éléments ne peut créer de nouveau sous-ensemble, comme on peut le voir dans le tableau 5.2. Toute fusion impliquant des unions d'ensembles pourrait mener à l'apparition de croyance sur des éléments non-interprétables par le robot.

m^{Ω}_{audio} \ m^{Ω}_{vision}	H	$\{H, O\}$	F	Ω
H	H	H	$\emptyset$	H
Ω	H	$\{H, O\}$	F	Ω

TABLE 5.2 – Intersections possibles entre les différents éléments focaux d'une fonction de croyance m^{Ω}_{audio} associée à une cellule de la grille évidentielle audio et d'une fonction de croyance m^{Ω}_{video} associée à une cellule de la grille évidentielle vidéo

A nouveau ici, on peut voir qu'il n'existe qu'un seul cas de conflit possible : lorsque la cellule concernée reçoit une BBA "Humain" de la part de la modalité audio, et une BBA "vide" de la part de la vision. En prenant en compte l'incertitude liée à la localisation de source audio, il semble naturel de porter la confiance sur la vision. En effet, l'audio ne donne que des candidats potentiels de cellules pouvant contenir de l'humain, tandis que la vision donne une information spécifique par rapport au contenu de la cellule.

Ainsi on peut utiliser pour créer la grille évidentielle de perception la règle de fusion suivante :

$$m_P^\Omega(S) = \left(m_{audio}^\Omega \textcircled{\cap} m_{vision}^\Omega\right)(S), \quad \forall S \subseteq \Omega, S \neq F \tag{5.15}$$

$$m_P^\Omega(F) = \left(m_{audio}^\Omega \textcircled{\cap} m_{vision}^\Omega\right)(F) + (m_{audio}^\Omega \textcircled{\cap} m_{vision}^\Omega)(\emptyset) \tag{5.16}$$

où m_P^Ω représente la distribution de masses de la grille de perception, résultante de la fusion au sein d'une cellule.

En procédant ainsi, les éléments focaux de la grille de perception restent les mêmes que ceux des grilles de vision et audio. La grille de perception contient l'ensemble des éléments perçus à un instant d'observation t donné. Elle peut être utilisée pour mettre à jour la grille construite lors des instants de perception précédents.

5.3.4 Fusion temporelle

Les capteurs en présence étant susceptibles d'être par moment défaillants (erreur de mesure laser, faux positif/négatif dans la détection d'humains, etc..) il est intéressant d'utiliser le savoir passé pour mettre à jour la grille à l'aide de l'information perçue. Ainsi les inconsistances lors de la mise à jour peuvent permettre de traiter des défauts capteur ou de tirer du sens de la scène qui entoure le robot. Mettre à jour la grille revient à fusionner la grille estimée à l'instant $t-1$ avec la grille à l'instant t. Pour ce faire, on propose d'utiliser un modèle de Markov crédibiliste, introduit par [RRP07], qui pose un formalisme sur les processus crédibilistes temporels. On commence donc par décrire le modèle de Markov caché (HMM, pour Hidden Markov Model) crédibiliste, puis on en décrira l'utilisation dans le cadre de ce travail.

5.3.4.1 Modèle de Markov caché crédibiliste

Le modèle de Markov caché crédibiliste est une extension du modèle de Markov probabiliste [Rab89 ; Mur02]. Le principe est le suivant : connaissant l'état estimé d'un système crédibiliste à l'instant t-1, et ayant à disposition une connaissance sur les croyances de transition d'un état à un autre entre deux instants successifs, ainsi qu'une observation sur l'état du système à l'instant t, on cherche à estimer l'état réel du système à l'instant t.

On considère un système dont les états réels possibles font partie de l'espace de discernement Θ quelconque de taille finie. On définit l'état estimé du système à l'instant t par une

distribution de masses $\widehat{m}_t^{\Theta}$. Un modèle de Markov caché crédibiliste se définit à l'aide de trois éléments :

- π_0^{Θ} est une distribution de masses, définie sur 2^{Θ}, représentant la croyance initiale sur l'état du système, avant la première observation.
- $m_{\text{trans}}^{\Theta}[S_i](S_j)$, $\forall S_i, S_j \in 2^{\Theta}$ représente la matrice de transition d'état. Elle contient la croyance en la transition d'un état crédibiliste S_i à l'instant $t-1$ à un état crédibiliste S_j à l'instant t. Elle est donc de taille $2^{|\Theta|} * 2^{|\Theta|}$.
- $\{\bar{m}_1^{\Theta}, \cdots, \bar{m}_t^{\Theta}\}$, est l'ensemble des observations successives faites sur le système aux différents instants d'observation. Chaque observation est représentée de manière crédibiliste par une distribution de masses (modèle capteur) définie sur Θ.

Comme pour le cas probabiliste, l'estimation de l'état du système $\widehat{m}_t^{\Theta}$ à l'instant t est un procédé en deux étapes. Tout d'abord, l'état crédibiliste du système à l'instant t est prédit, donnant une distribution de masses $\widetilde{m}_t^{\Theta}$:

$$\underbrace{\widetilde{m}_t^{\Theta}(S_j)}_{\text{Masse prédite à t}} = \sum_{S_i \subseteq \Theta} \left(\underbrace{\widehat{m}_{t-1}^{\Theta}(S_i)}_{\text{Masse estimée à t-1}} \cdot \underbrace{m_{\text{trans}}^{\Theta}[S_i](S_j^t)}_{\substack{\text{Croyance de} \\ \text{transition d'états}}} \right), \quad \forall S_j \subseteq \Theta \tag{5.17}$$

Ceci revient à une combinaison conjonctive entre l'estimée à l'instant $t-1$ et la matrice de transition, et permet de transférer la croyance de l'estimée à l'instant $t-1$ vers l'hypothèse la plus crédible à l'instant t, connaissant les tendances d'évolution du système entre deux instants successifs.

La prédiction $\widetilde{m}_t^{\Theta}$ est ensuite corrigée à l'aide du modèle capteur des observations à l'instant t. A nouveau, une combinaison conjonctive est faite entre la prédiction et l'observation, pour obtenir l'estimée finale :

$$\underbrace{\widehat{m}_t^{\Theta}(S_j)}_{\text{Masse estimée à t}} = \sum_{\substack{S_k \cap S_i = S_j \\ S_k, S_i \subseteq \Theta}} \left(\underbrace{\widetilde{m}_t^{\Theta}(S_k)}_{\text{Masse prédite à t}} \cdot \underbrace{\bar{m}_t^{\Theta}(S_i)}_{\substack{\text{Modèle capteur} \\ \text{des observations à t}}} \right), \quad \forall S_j \subseteq \Theta \tag{5.18}$$

Ce formalisme présente des avantages : l'étape explicite de prédiction ouvre plusieurs possibilités, par exemple pour l'apprentissage de l'évolution de l'état d'une cellule. Qui plus est, l'étape de correction consiste en une fusion classique de fonctions de croyance : elle est donc adaptable aux spécificités du système considéré.

5.3.4.2 HMM crédibiliste : application à l'espace de discernement $\Omega = \{H, O, F\}$

Ici, le HMM crédibiliste est appliqué à chacune des cellules des grilles évidentielles, puisqu'on estime la distribution de masses de toutes les cellules de la grille simultanément.
Dans un premier temps, on conserve l'espace de discernement $\Omega = \{H, O, F\}$. Chaque nouvelle grille de perception (définie au paragraphe 5.3.3), correspond à une nouvelle observation $\bar{m}_t^\Omega$, qui permet de mettre à jour la grille (équation 5.18). Il reste donc deux éléments à définir :

— La distribution de masses initiale π_0^Ω, soit l'initialisation de la grille finale, avant toute perception du robot. N'ayant pas d'information sur l'état du monde avant la perception, on initialise chaque cellule de la grille avec une distribution de masses vide ($\pi_0^\Omega(\Omega) = 1$), qui représente l'ignorance totale.

— La matrice de transition entre deux instants successifs m_{trans}^Ω. Chaque cellule étant considérée de manière indépendante de ses voisines, on ne peut pas prendre en compte l'état des cellules voisines pour prédire l'état d'une cellule à l'instant suivant. Ainsi, l'unique transition que l'on peut faire sur une cellule est un affaiblissement, soit une diminution de la croyance sur toutes les hypothèses autres que Ω. Ceci permet de tendre vers l'ignorance dans le temps dans le cas où on cesse d'observer l'état de la cellule.
Un affaiblissement se traduit mathématiquement par :

$$\widetilde{m}_t^\Omega(S_i) = \widehat{m}_{t-1}^\Omega(S_i) \cdot e^{-\gamma} \quad \forall S_i \subseteq \Omega \tag{5.19}$$

$$\widetilde{m}_t^\Omega(\Omega) = 1 - \sum_{S_i \subseteq \Omega} \left(\widetilde{m}_t^\Omega(S_i)\right) \tag{5.20}$$

Donc, sans observation supplémentaire, la distribution de masses d'une cellule tendra dans le temps vers une distribution de masses vide (soit l'ignorance complète). A partir d'un tel affaiblissement, on peut écrire la matrice de transition suivante :

$$m_{trans}^\Omega[S_i](S_j) = 0, \quad \forall i \neq j, S_j \neq \Omega \tag{5.21}$$

$$m_{trans}^\Omega[S_i](S_i) = e^{-\gamma}, \quad S_i \neq \Omega \tag{5.22}$$

$$m_{trans}^\Omega[S_i](\Omega) = 1 - e^{-\gamma}, \quad \forall S_i \neq \Omega \tag{5.23}$$

$$m_{trans}^\Omega[\Omega](\Omega) = 1 \tag{5.24}$$

Avec ces éléments, l'étape de prédiction est bien définie. Pour la correction, on utilise une fusion conjonctive entre l'observation à l'instant t, et la prédiction faite $\widetilde{m}_t^\Omega$ à partir de l'estimée à l'instant $t-1$. L'observation $\bar{m}_t^\Omega$ est représentée par la grille de perception à l'instant t : $m_{P,t}^\Omega$.

Toutefois, comme on l'a dit, avec un affaiblissement en guise de prédiction, il n'est pas possible de prévoir un changement au sein d'une cellule. Ainsi, si une cellule a changé d'état réel entre l'instant $t-1$, et l'instant t, une combinaison conjonctive entre la prédiction et l'observation pourra provoquer du conflit. Or, le conflit est un élément absorbant de la combinaison conjonctive : l'intersection de $\emptyset$ avec n'importe quel élément de 2^Ω est $\emptyset$ lui-même. Ceci signifie qu'à partir du moment où de la croyance est portée sur le conflit, les combinaisons conjonctives postérieures feront de la distribution de masses concernée une distribution de masses où toute

la croyance est concentrée sur le conflit ($\mathrm{m}(\emptyset) = 1$). Ceci n'est pas acceptable : le conflit doit donc être géré à chaque étape de la fusion. Pour ce faire, il convient d'en analyser les sources possibles.
Le conflit apparaîtra :

- Si une cellule se vide de son contenu (quelque chose a bougé) entre l'instant $t-1$ et l'instant t, auquel cas la prédiction gardera une croyance forte sur la présence d'un objet au sein de cette cellule (hypothèse H, ou $\{H, O\}$), tandis que l'observation donnera une croyance forte sur l'hypothèse F (espace navigable).
- Si une cellule se remplit (quelque chose vient d'arriver sur la cellule) entre l'instant $t-1$ et l'instant t. Dans ce cas la prédiction gardera une croyance forte sur F, tandis que l'observation donnera une croyance forte sur la présence d'un objet.

Sans modélisation explicite de la dynamique de la scène dans l'espace de discernement Ω, on choisit de donner la priorité à l'observation au temps t ($m^{\Omega}_{P,t}$) : en effet, décider d'accorder la croyance du conflit à l'hypothèse avancée par la prédiction ne permet pas la mise à jour de la grille.

On utilise ainsi la combinaison conjonctive modifiée suivante :

$$\widehat{m}^{\Omega}_t(S) = \left(\widetilde{m}^{\Omega}_t \textcircled{\cap} m^{\Omega}_{P,t}\right)(S) + \sum_{\substack{A \subseteq \Omega \\ A \cap S = \emptyset}} \left(m^{\Omega}_{P,t}(S) \cdot \widetilde{m}^{\Omega}_t(A)\right) \tag{5.25}$$

$$\widehat{m}^{\Omega}_t(\emptyset) = 0 \tag{5.26}$$

Cette première proposition permet de faire face à certains problèmes, comme la perte de détection d'un humain dans la grille évidentielle vidéo : à l'instant $t-1$, un être humain est détecté dans la vidéo et replacé dans une cellule. Le laser perçoit aussi un objet dans cette cellule. Ainsi, à l'instant $t-1$, une BBA "humain" est attribuée à cette cellule. A l'instant t, le détecteur d'humain fait défaut et l'humain n'est plus détecté. Cependant le laser continue de percevoir un objet. Dans ce cas, la combinaison conjonctive entre la prédiction (qui prévoit la présence d'un humain dans cette cellule, avec l'hypothèse H), et l'observation (qui donne pour cette cellule une distribution de masses "objet ou humain", avec un élément focal $\{H, O\}$), permet de garder une croyance forte sur H pendant plusieurs itérations (l'intersection de $\{H, O\}$ et H étant H).

Cependant la modélisation proposée ici présente des limites : une cellule contenant un humain en train de se déplacer recevra a priori la même distribution de masses qu'une cellule contenant un humain statique, car l'espace de discernement utilisé jusqu'à présent ne permet pas de différencier les deux cas (l'hypothèse H correspondant à un humain statique ou mobile). On propose donc une deuxième implémentation du HMM crédibiliste prenant cette fois ci en compte la dynamique de la scène.

5.3.4.3 HMM crédibiliste : modélisation de la dynamique de la scène

On souhaite pouvoir différencier entre l'environnement statique du robot (humain ou objet immobile) et l'environnement mobile (humain ou objet en mouvement). Pour modéliser explicitement le mouvement dans la scène, on se propose d'introduire un nouvel espace de discernement $\Theta = \{H_s, H_m, O_s, O_m, F\}$, inspiré de [MCB11 ; Kur+15] :

— H_s : présence d'un humain statique au sein de cette cellule,
— H_m : présence d'un humain mobile au sein de cette cellule,
— O_s : présence d'un objet statique au sein de cette cellule,
— O_m : présence d'un objet mobile au sein de cette cellule,
— F : espace libre navigable.

Pour pouvoir travailler avec cet espace de discernement, il est nécessaire de projeter les éléments de l'espace de discernement $\Omega = \{H, O, F\}$ dans 2^Θ par opération dite "de raffinement" de l'espace de discernement. On définit le raffinement Γ de la manière suivante :

$$\begin{array}{rccc} \Gamma : & \Omega & \rightarrow & 2^\Theta \\ & \{H\} & \mapsto & \{H_s, H_m\} \\ & \{O\} & \mapsto & \{O_s, O_m\} \\ & \{F\} & \mapsto & \{F\} \end{array} \tag{5.27}$$

Ce raffinement signifie qu'un humain (resp. un objet) est soit statique, soit mobile. Pour obtenir la grille de perception $m^\Theta_{P,t}$ sur l'espace de discernement Θ on définit l'opération suivante :

$$\begin{array}{ccc} 2^\Omega & \rightarrow & 2^\Theta \\ m^\Omega_{P,t}(B) & \mapsto & m^\Theta_{P,t}(\Gamma(B)), \quad \forall B \subseteq \Omega \end{array} \tag{5.28}$$

Au terme de ce raffinement, on obtient une grille de perception définie sur Θ qui permet d'améliorer l'utilisation du HMM. Il n'est pas possible d'améliorer la partie prédiction (la matrice de transition reste inchangée).
Concernant la correction par les observations, le conflit provoqué lors de la combinaison conjonctive peut cette fois être exploité. Pour les deux cas de conflits entre deux instants successifs (si une cellule se vide ou inversement si elle se remplit), on peut donc transférer le conflit de la manière suivante :

— Cellule qui se vide (la grille de perception donne une BBA "espace libre navigable", lorsque la prédiction donne une BBA "objet") : on choisit de donner la priorité à l'observation, pour les mêmes raisons que celles citées précédemment.
— Cellule qui se remplit (la grille de perception donne une BBA "objet", lorsque la prédiction donne une BBA "espace libre navigable") : on considère qu'un objet mobile vient d'entrer dans cette cellule, et le conflit généré est transféré sur l'hypothèse mobile sous-jacente.

Ceci correspond à la combinaison conjonctive modifiée suivante :

$$\widehat{m}_t^{\Theta_t}(A) = \left(\widetilde{m}_t^{\Theta} \textcircled{\cap} m_{P,t}^{\Theta}\right)(A) + \sum_{\substack{A=(B\cap\{H_m,O_m\}) \\ B\subseteq\Theta\setminus F}} \left(\widetilde{m}_t^{\Theta}(F)\cdot m_{P,t}^{\Theta}(B)\right), \quad \forall A \subseteq \{H_m, O_m\} \tag{5.29}$$

$$\widehat{m}_t^{\Theta}(F) = \left(\widetilde{m}_t^{\Theta} \textcircled{\cap} m_{P,t}^{\Theta}\right)(F) + \sum_{A\subseteq\Theta\setminus F} \left(\widetilde{m}_t^{\Theta}(A)\cdot m_{P,t}^{\Theta}(F)\right) \tag{5.30}$$

$$\widehat{m}_t^{\Theta}(A) = \left(\widetilde{m}_t^{\Theta} \textcircled{\cap} m_{P,t}^{\Theta}\right)(A), \quad \forall A \quad \not\subseteq \{H_m, O_m\}, A \neq F \tag{5.31}$$

En procédant ainsi, les éléments mobiles de la scène seront bien détectés. Cependant, le système n'ajoute jamais de croyance sur les hypothèses H_s et $\{H_s, O_s\}$: une fois qu'une cellule contient un élément qualifié "mobile", il ne peut plus devenir "statique", même s'il ne bouge pas, sauf dans le cas où le taux d'affaiblissement γ varie selon les hypothèses considérées dans la matrice de transition $\widetilde{m}^{\Omega}_{\text{trans}}[S_i](S_j)$. Qui plus est, le système risque d'augmenter artificiellement la croyance sur l'hypothèse mobile correspondante, même une fois que l'élément en question se sera immobilisé : par exemple si à un instant t, un humain entre dans une cellule, celle-ci recevra une distribution de masses "humain mobile" (H_m). Si l'humain s'arrête et continue d'être détecté, la perception fournira une distribution de masses "humain" ($\{H_s, H_m\}$). L'intersection entre ces deux hypothèses étant H_m, il devient impossible de qualifier l'humain comme étant statique ($\{H_s\}$).

Pour pallier ce problème, on propose un test d'immobilité. Le principe est le suivant : si une cellule a été occupée pendant plus d'un temps pré-défini, toute la croyance de ses hypothèses mobiles est transférée sur les hypothèses statiques correspondantes. On définit le degré d'occupation d'une cellule comme suit :

$$Occ = \widehat{bel}_t^{\Theta}(\{H_m, H_s, O_m, O_s\}) = \sum_{S\subseteq\{H_m,H_s,O_m,O_s\}} \left(\widehat{m}_t^{\Theta}(S)\right) \tag{5.32}$$

Ainsi, si $Occ > \delta$ pendant une durée Δt, avec δ et Δt prédéfinis, on procède aux transferts de croyance suivants :

$$\widehat{m}_t^{\Theta}(\{H_m\}) \mapsto \widehat{m}_t^{\Theta}(\{H_s\}) \tag{5.33}$$

$$\widehat{m}_t^{\Theta}(\{H_m, O_m\}) \mapsto \widehat{m}_t^{\Theta}(\{H_s, O_s\}) \tag{5.34}$$

$$\widehat{m}_t^{\Theta}(\{H_m, O_m, H_s, O_s\}) \mapsto \widehat{m}_t^{\Theta}(\{H_s, O_s\}) \tag{5.35}$$

De cette manière, si un objet précédemment mobile s'arrête soudainement et reste détecté, au bout d'un temps choisi il sera de nouveau considéré comme statique par le robot.

5.4 Résultats

Le système présenté dans ce chapitre a d'abord été implémenté en simulation avant d'être implémenté sur un robot réel. Réaliser un simulateur permet de démontrer en profondeur le fonctionnement du système, en contrôlant tous les paramètres. Ainsi, on illustre dans un premier temps l'intégralité du système (les deux espaces de discernement) sur un exemple simulé, puis les résultats obtenus sur robot réel.

5.4.1 Résultats en simulation

5.4.1.1 Présentation du simulateur

Grille d'occupation :

Pour la simulation, on crée une grille factice d'occupation représentant l'environnement dans lequel le robot évolue. Cet environnement est illustré par la figure 5.12.

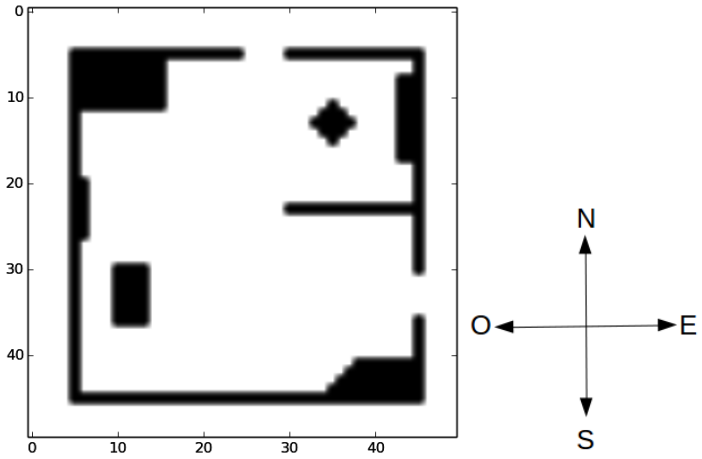


FIGURE 5.12 – Grille d'occupation utilisée pour la simulation : le blanc représente l'espace libre navigable, et le noir les objets statiques de l'environnement

La grille d'occupation est une grille de 50x50 pixels (soit 25m^2, si on considère qu'une cellule est un carré de 10cm de côté) représentant une grande pièce de vie, contenant deux portes. Les objets peuvent représenter différents types d'obstacles pour le robot : table, murs de la pièce, plan de travail, étagères, etc...

Simulation des capteurs :

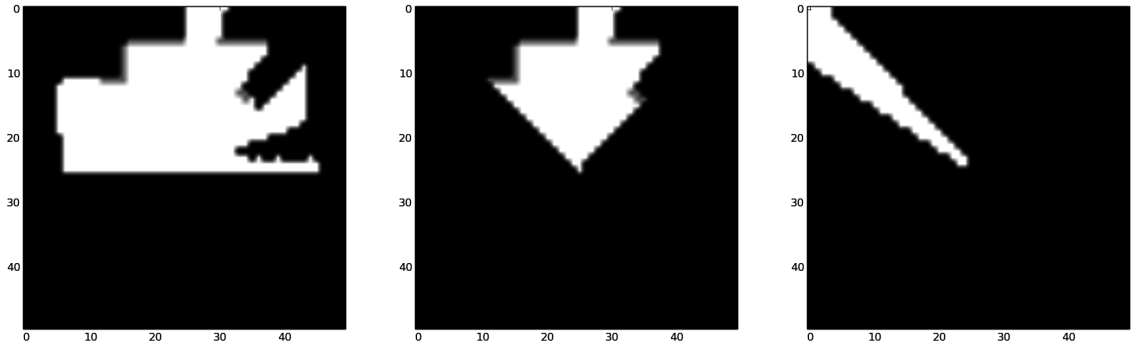
Les champs de perception des capteurs simulés sont illustrés par la figure 5.13 :

— **Laser (figure 5.13 (a)) :** Le laser a un champ de perception de 180 °. La simulation ayant principalement pour but l'illustration du système, on considère que le laser est

parfait : tous les obstacles sont détectés à la bonne distance à chaque instant d'observation

— **RGB-D (figure 5.13 (b)) :** Le capteur RGB-D a un champ de perception de 90 °. Il est lui aussi considéré parfait : tous les humains sont détectés comme tels.

— **Audio (figure 5.13 (c)) :** Le champ de perception audio est considéré comme étant le même que celui du laser : 180 °. Pour la simulation, on utilise un cône d'incertitude large, de taille angulaire 20 °, centré sur l'angle estimé par la localisation



(a) Laser (b) Capteur RGB-D (c) Cône audio

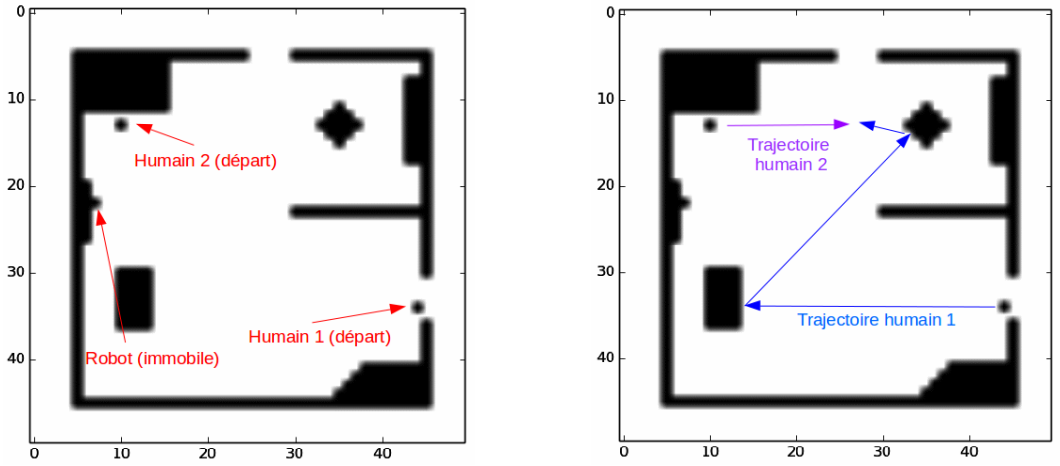
FIGURE 5.13 – Champs de perception des capteurs simulés : Laser, RGB-D, et audio. Sur cet exemple, le robot se trouve au centre de la grille d'occupation, tourné vers le nord.

Robot et humains :

Dans les scénarios présentés, le robot est considéré comme immobile : ceci permet de s'affranchir de la problématique de localisation et de se concentrer sur l'illustration de la construction des grilles évidentielles.
Les humains sont considérés comme des obstacles ponctuels (une seule cellule de la grille occupée), pouvant se déplacer d'une seule cellule par temps d'observation. Ils peuvent être silencieux ou émettre des sons.

5.4.1.2 Description du scénario

Dans le scénario, le robot est témoin d'une interaction entre deux humains. Il est placé contre un mur de la pièce, en position d'observation (figure 5.14 (a)). Deux humains sont présents. Au début du scénario, l'humain 1 est visible par le capteur RGB-D, tandis que l'humain 2 n'est visible que par le capteur laser. Tous deux sont silencieux. Dans un premier temps, seul l'humain 1 bouge, selon la trajectoire indiquée dans la figure 5.14 (b). Puis les deux humains se rejoignent au centre de la pièce. Au cours des déplacements, une conversation a lieu : les deux humains parlent successivement avec des temps de pause.



(a) Position initiale (b) Trajectoire des humains

FIGURE 5.14 – Scénario proposé pour la simulation de construction de grilles évidentielles

5.4.1.3 Réalisation du scénario et construction des grilles : cas de l'espace de discernement $\Omega = \{H, O, F\}$

On commence par réaliser le scénario, avec l'espace de discernement initial : $\Omega = \{H, O, F\}$. On se concentre ici sur l'élaboration de la grille de perception et la grille finale, à différents instants clés du scénario. Pour rendre la lecture des résultats plus facilement compréhensible, on montre pour chaque cellule de la grille l'hypothèse ayant reçu le plus de croyance. Le code couleur est illustré par la figure 5.15

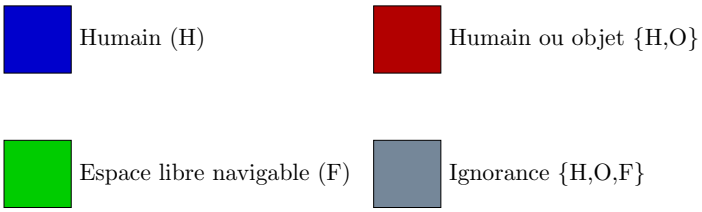


FIGURE 5.15 – Code couleur pour les figures sur la réalisation du scénario avec l'espace de discernement $\Omega = \{H, O, F\}$.

Quatre moments différents du scénario sont représentés par la figure 5.16, chacun permettant d'illustrer un point clé de la fusion :

— **t = 0 : première perception.** Le robot commence à percevoir et les 2 humains sont silencieux. L'humain 1 se trouve dans le champ de vision du capteur RGB-D : il est donc détecté comme tel et reçoit une distribution de masses "humain". L'humain 2 n'est que dans le champ de vision du capteur laser. Il reçoit donc la même distribution de masses que le reste de l'environnement, "humain ou objet". Comme le robot vient de

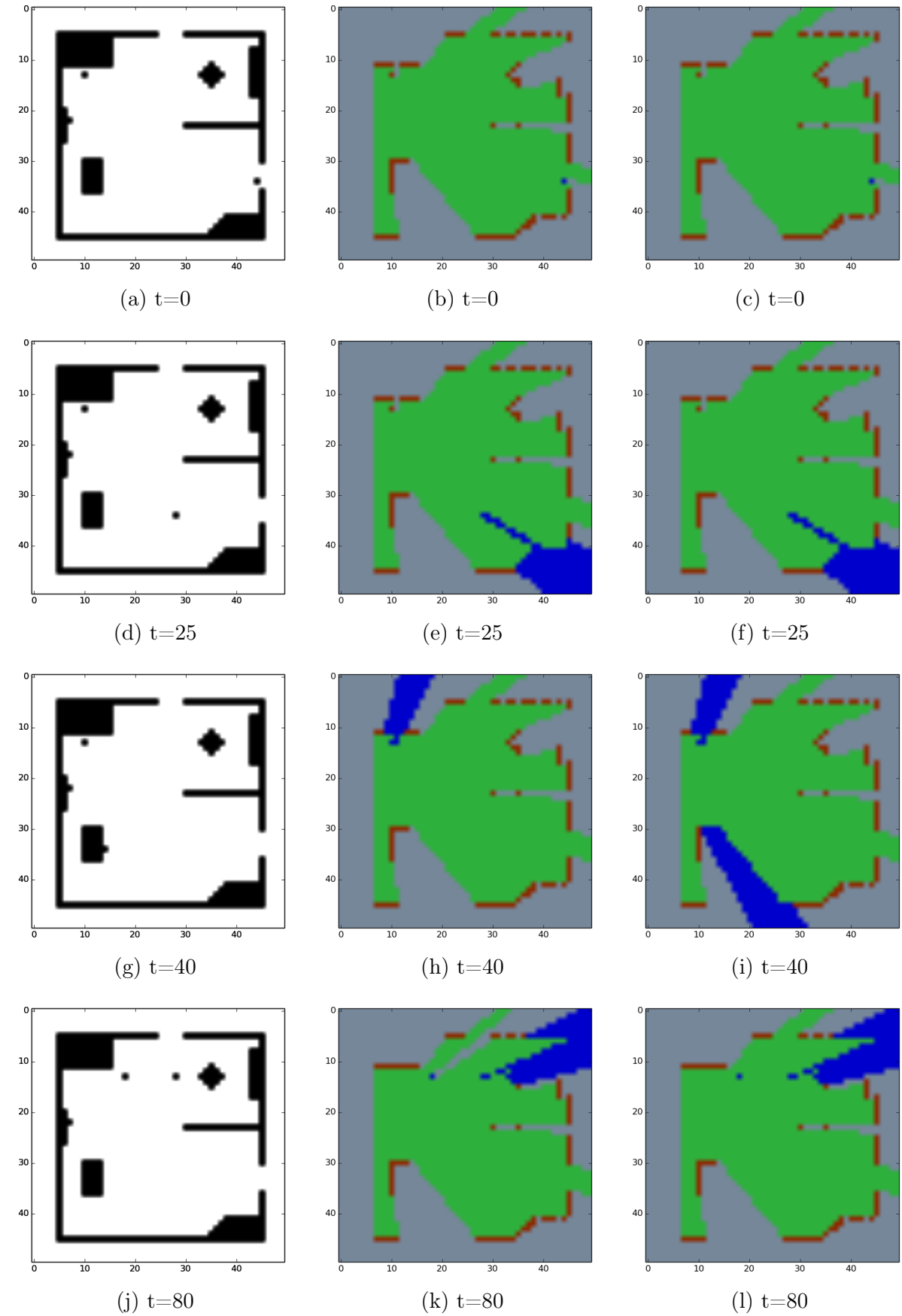
commencer à percevoir, aucune fusion temporelle n'a eu lieu et les grilles de perception et finale sont les mêmes.

— **t = 25 : l'humain 1 entame le dialogue et devient source sonore, en se déplaçant.** Le cône a bien le comportement attendu : lors de la fusion avec l'espace libre navigable de la grille de vision, on donne la priorité à la vision et les cellules reçoivent une distribution de masses "espace libre navigable". L'humain, qui continue à être détecté comme tel reçoit toujours une distribution de masses "humain", ainsi que les objets de l'environnement faisant partie du cône : ceci est dû à la nature de la fusion conjonctive. Les cellules contenant des objets de l'environnement reçoivent une distribution de masses "humain ou objet" lors de la construction de la grille de vision. Lors de la fusion pour créer la grille de perception, cette distribution de masses est fusionnée avec une distribution "humain" du cône audio. L'intersection de $\{H, O\}$ et H étant H, les objets reçoivent une distribution "humain". Ceci permet aussi de respecter un principe de prudence : il n'y aucune garantie que la source du son entendu est une personne détectée visuellement : il est donc acceptable que tout objet perçu dans le cône audio soit considéré comme un humain potentiel. Pour finir, on note que le cône audio dépasse les limites de la pièce : ceci est dû au fait que sans information a priori sur les objets de l'environnement (mur ou mobilier), il n'est pas possible de savoir quel type d'obstacle arrête le son.

— **t = 40 : l'humain 1 a cessé de parler et l'humain 2 a commencé à lui répondre.** La fusion audiovisuelle pour l'humain 2 est similaire à celle de t=25. Cependant, on note sur la grille finale que le cône de l'humain 1 persiste pendant un moment. Ceci est dû à l'étape de prédiction dans la fusion temporelle : les masses précédemment perçues s'affaiblissent doucement, ce qui fait persister le cône pendant quelques instants d'observation.

— **t = 80 : changement de perception.** L'humain 1 s'est positionné au centre de la pièce, en train de parler. L'humain 2 vient d'entrer dans le champ de vision du capteur RGB-D. Il reçoit donc une distribution de masses "humain", puisqu'il est visuellement détecté. On note que l'espace que l'humain cache au robot reçoit une distribution vide dans la grille de perception mais une distribution "espace libre navigable" dans la grille finale. Ceci vient à nouveau de la fusion temporelle : le robot garde en mémoire pendant un instant que ces cellules contiennent de l'espace libre, et lorsque l'humain les cache, continue de leur attribuer une telle distribution.

Le système agit donc comme attendu, et les différentes fusions illustrent la réalité de la scène de manière prudente.



(a) t=0 (b) t=0 (c) t=0

(d) t=25 (e) t=25 (f) t=25

(g) t=40 (h) t=40 (i) t=40

(j) t=80 (k) t=80 (l) t=80

FIGURE 5.16 – Réalisation du scénario, espace de discernement $\Omega = \{H, O, F\}$: grilles d'occupation, de perception et finale au cours du temps

5.4.1.4 Réalisation du scénario et construction des grilles : cas de l'espace de discernement $\Theta = \{H_s, H_m, O_s, O_m, F\}$

On réalise le même scénario dans l'espace de discernement augmenté. Dans ce cas, on peut observer une quantité plus grande de phénomènes au cours du temps. Le code couleur pour les figures correspondant à cet espace de discernement est donné par la figure 5.17.

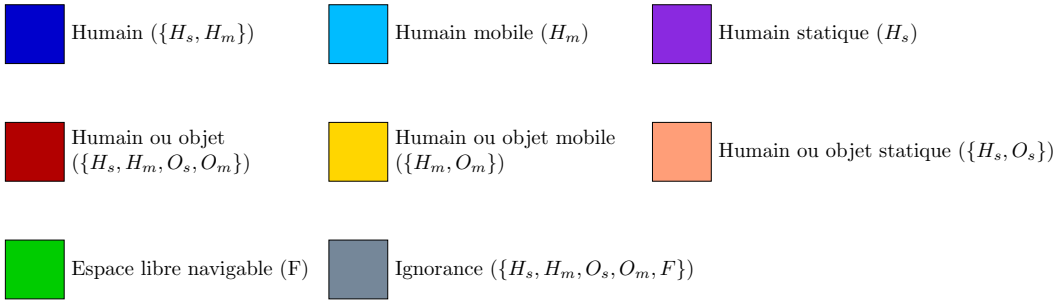
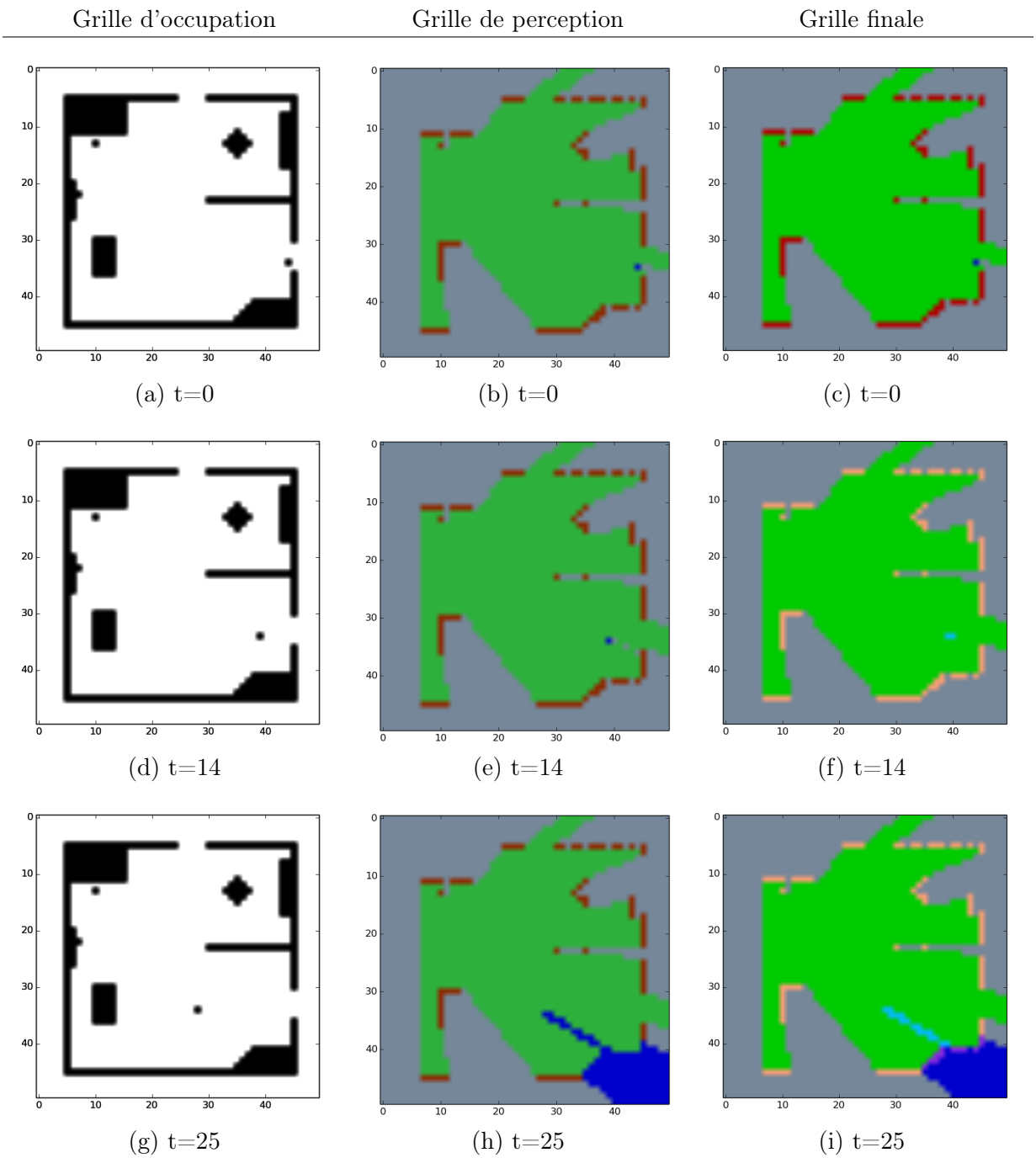


FIGURE 5.17 – Code couleur pour les figures sur la réalisation du scénario avec l'espace de discernement $\Theta = \{H_s, H_m, O_s, O_m, F\}$.

La figure 5.18 montre les grilles de perception et finale à certains instants importants de la réalisation du scénario :

- **t = 0 : début de la perception.** On retrouve ici les mêmes grilles que dans le cas $\Omega = \{H, O, F\}$. Les objets de l'environnement, ainsi que l'humain 2 reçoivent une distribution "humain ou objet" ($\{H_s, H_m, O_s, O_m\}$), et l'humain 1 reçoit une distribution "humain" ($\{H_s, H_m\}$). Comme aucune intégration dans le temps n'a eu lieu pour le moment, il est impossible de différencier entre les objets mobiles ou statiques de l'environnement, et la grille de perception est la même que la grille finale.
- **t = 14 : les objets statiques sont considérés comme tels.** Les objets de l'environnement n'ayant pas bougé depuis un certain temps dans la grille finale reçoivent une distribution "humain ou objet statique" ($\{H_s, O_s\}$), grâce au test de immobilité. Par ailleurs l'humain 1, en train de se diriger vers une table, reçoit une distribution de masses "humain mobile", lors de l'étape de correction par les observations du HMM crédibiliste : en effet, il remplit à chaque nouvel instant une cellule considérée à l'instant précédent comme de l'espace libre navigable.
- **t = 25 : l'humain 1 parle pour la première fois, tout en se déplaçant.** La grille de perception est la même que dans l'espace de discernement $\Omega = \{H, O, F\}$, les capteurs ne pouvant pas donner d'information supplémentaire. Cependant la fusion temporelle améliore la grille finale : en effet, les objets statiques présents dans le cône audio reçoivent cette fois une distribution "humain statique" (H_s), cependant que l'humain 1 reçoit lui bien une distribution "humain mobile" (H_m). Ceci est dû à la fusion conjonctive lors de la correction par les observations.

Deux étapes supplémentaires sont montrées par la figure 5.19 :



(a) t=0 (b) t=0 (c) t=0

(d) t=14 (e) t=14 (f) t=14

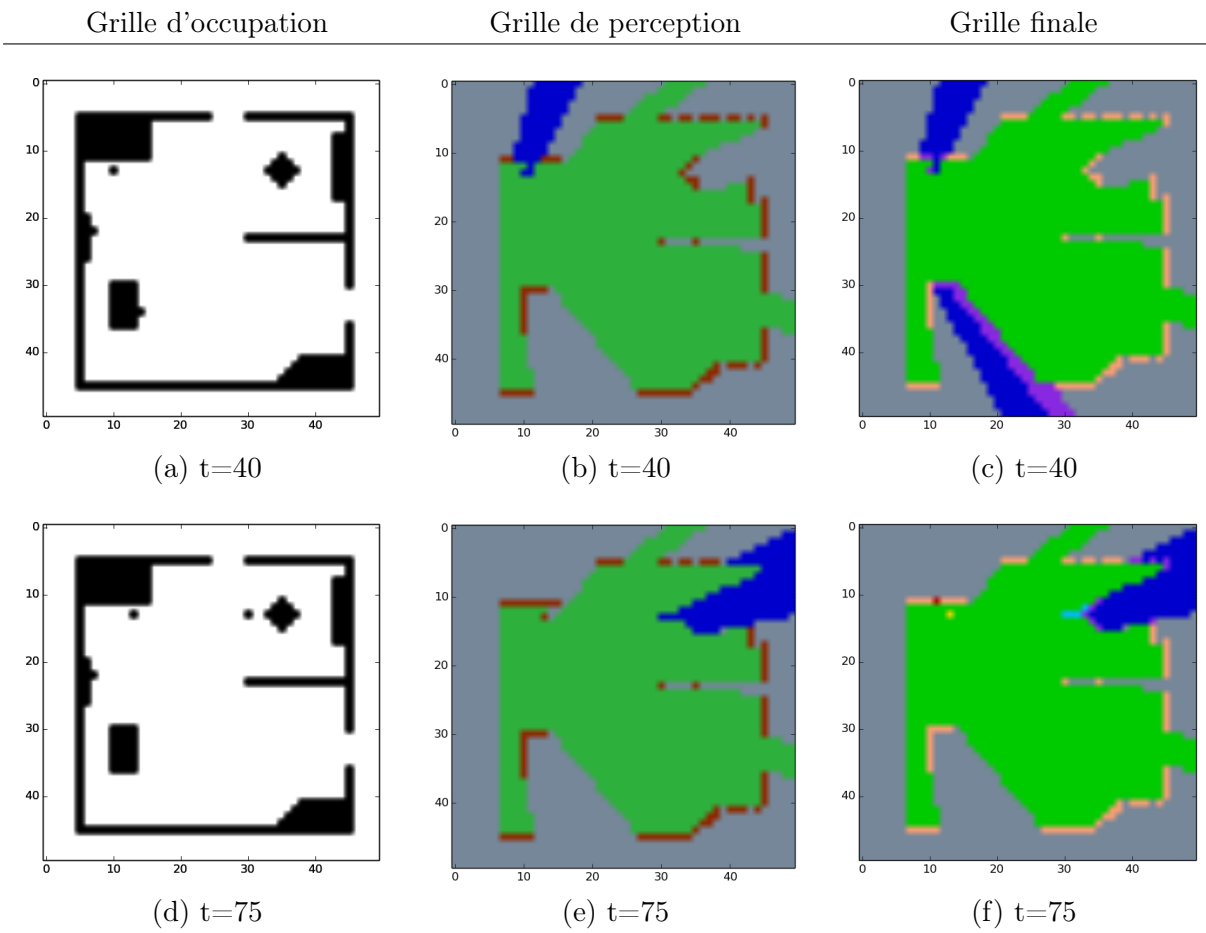
(g) t=25 (h) t=25 (i) t=25

FIGURE 5.18 – Réalisation du scénario, espace de discernement $\Theta = \{H_s, H_m, O_s, O_m, F\}$: grilles d'occupation, de perception et finales au cours du temps

— **t = 40 : l'humain 2 se met à parler.** Comme dans le cas $\Omega = \{H, O, F\}$, lorsque l'humain 2 se met à parler, on observe une persistance du cône audio de l'humain 1, dont une partie seulement a déjà subi le test d'immobilité et reçoit une distribution "humain statique" (H_s), tandis que la deuxième partie du cône n'existe pas encore depuis assez longtemps pour être considérée comme statique, et reçoit une distribution "humain" ($\{H_s, H_m\}$). De la même manière, c'est le premier instant d'observation où l'humain 2 est entendu, il ne peut donc être considéré ni comme mobile, ni comme statique.

— **t = 75 : L'humain 2 se dirige vers l'humain 1, sans être encore perçu par le capteur RGB-D.** Il reçoit donc une distribution "humain ou objet mobile" ($\{H_m, O_m\}$). Le point statique que l'humain 2 cachait au robot jusque là reçoit pour l'instant une distribution "humain ou objet" ($\{H_s, H_m, O_s, O_m\}$) car il n'a pas encore subi le test d'immobilité. Concernant l'humain 1, il vient de s'arrêter et n'est pour l'instant pas considéré comme statique. Il faut pour cela attendre encore quelques observations.

L'utilisation de l'espace de discernement $\Theta = \{H_s, H_m, O_s, O_m, F\}$ permet donc d'améliorer la représentation de l'environnement du robot en différenciant dès que possible entre les objets statiques de l'environnement et les objets mobiles.



(a) t=40 (b) t=40 (c) t=40

(d) t=75 (e) t=75 (f) t=75

FIGURE 5.19 – Réalisation du scénario, espace de discernement $\Theta = \{H_s, H_m, O_s, O_m, F\}$: grilles d'occupation, de perception et finales au cours du temps

5.4.2 Résultats sur le robot réel

Nous présentons les résultats de l'implémentation du système sur un robot réel. Les manipulations ont été réalisées exclusivement avec l'espace de discernement $\Omega = \{H, O, F\}$ pour deux raisons : le coût de calcul avec l'espace Θ est trop élevé à l'heure actuelle, et le noeud ROS utilisé pour la localisation du robot ne permet pas de localiser le robot avec une précision assez grande pour prendre en compte la dynamique de la scène de manière assez robuste. La section est divisée de la manière suivante : les conditions d'expérimentations et les contraintes techniques sont tout d'abord présentées, puis deux expériences sont réalisées pour illustrer la construction des grilles. Finalement les performances de l'algorithme utilisé sont détaillées.

5.4.2.1 Conditions d'expérimentation

Les manipulations ont eu lieu au sein de la salle présentée dans le chapitre 4, dont l'organisation est rappelée brièvement dans la figure 5.20

(a) Vue 1 (b) Vue 2

FIGURE 5.20 – Salle d'expérimentation robotique

La grille d'occupation correspondant à cette pièce est rappelée dans la figure 5.21 (noir : obstacle statiques, blanc : espace libre navigable). La résolution de la grille est de 10cm (chaque cellule est un carré de 10cm par 10cm).

5.4.2.2 Contraintes techniques

Le robot utilisé pour les expériences est le même que dans le chapitre 4 : Q.bo. La création et la fusion de grilles évidentielles sont coûteuse en ressources de calcul. Le robot n'étant pas équipé d'un processeur puissant (Intel Core i3 basse consommation), on déporte le calcul et la création des grilles en temps réel sur un ordinateur connecté en Wi-Fi sur le même réseau que le robot, disposant de ressources de calcul supérieures (Processeur Intel Core i7-4710MQ Quad Core). Le logiciel ROS (Robot Operating System) offre des solutions de mise en réseau des machines simples permettant à tout ordinateur connecté au même réseau que Q.bo de

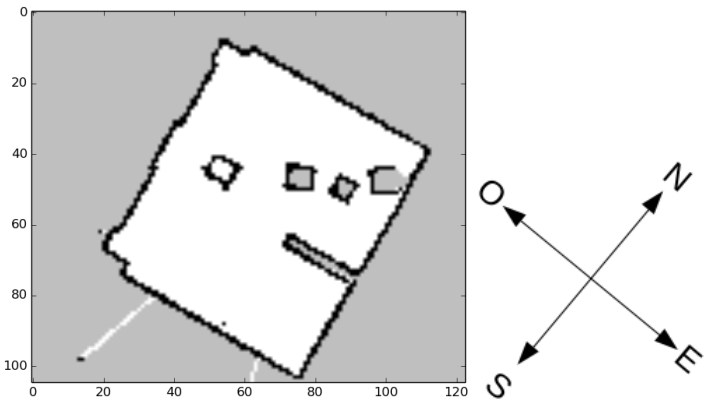


FIGURE 5.21 – Grille d'occupation représentant la salle d'expérimentation

partager les informations transitant au sein de ROS. On peut donc créer les grilles en temps réel pendant le mouvement du robot.

Pour se localiser dans la grille d'occupation, on utilise le paquet ROS "amcl" (pour "A Monte Carlo Localization"). Ceci permet au robot de recaler à chaque instant d'observation les grille construites dans son environnement.

Pour rendre la construction des grilles robuste au mouvement du robot, on utilise les mêmes détecteurs que dans le chapitre 4 : les humains sont équipés de gilets de couleur, dont la couleur est apprise en ligne de la même manière que dans le chapitre 3, ainsi que de haut parleurs qu'ils peuvent mettre en marche à tout moment pour émettre du bruit plus facilement localisable par le robot.

Pour illustrer la création de grilles, on propose de téléguider le robot à l'aide d'une manette de jeu vidéo. On se limite ici à l'utilisation de l'espace de discernement $\Omega = \{H, O, F\}$, et on propose deux scénarios différents.

5.4.2.3 Expérience 1 : translation du robot vers un humain statique silencieux

Dans ce scénario, le robot avance vers un humain en ligne droite, sans obstacle entre les deux. Le robot se trouve à 4m de l'humain au début du scénario, puis est arrêté lorsqu'il atteint une distance d'un mètre environ de l'être humain. La trajectoire est représentée par la figure 5.22

L'humain est silencieux, et présent dès la première observation dans le champ de vision du capteur RGB-D. Ainsi, seules les modalités vidéos et laser apportent de l'information, tandis que la grille audio sera toujours une grille contenant des distributions de masses vides.

Un exemple de construction des différentes grilles, lorsque le robot se trouve à grande distance (environ 4 mètres) de l'humain, est illustré par la figure 5.23.

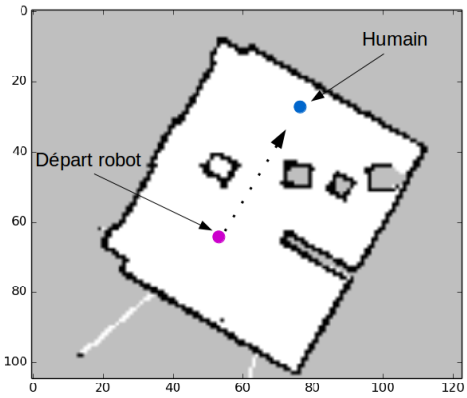


FIGURE 5.22 – Trajectoire suivie par le robot lors du scénario

Sur cette figure, l'ensemble des grilles est visible : la grille laser contient les éléments de l'environnement détectés par ce capteur. Chaque cellule contenant un impact laser reçoit une distribution de masses "humain ou objet" ($\{H, O\}$). Dans le même temps, la détection vidéo donne une présence humain replacée dans la grille vidéo sous la forme d'une distribution de masses "humain" (H, point bleu sur la grille). On choisit de donner une "épaisseur" à la détection d'humain en attribuant une distribution de masses "humain" aux voisins de la cellule où la détection a eu lieu. Ceci permet de faire face à des imprécisions légères de replacement des observations vidéos dans la grille.

La grille audio n'apporte comme prévu pas d'information, car aucune source sonore n'a été localisée. Elle est donc constituée intégralement de distributions de masses vides. La distribution de masses vide étant l'élément neutre de la fusion conjonctive, la grille de perception est égale à la grille de vision.

Finalement la grille de perception est fusionnée avec la grille finale affaiblie à $t-1$. Comme il y a peu de changement au sein de la scène, la fusion temporelle apporte peu d'information. Toutefois on peut noter deux effets :

— A l'instant $t-1$, l'humain n'est pas détecté dans la grille vidéo. Lors de la fusion temporelle, la priorité est bien donnée à l'observation pour ajouter l'humain à la grille finale à l'instant t,

— Certaines imperfections du lancé de rayon sont visibles dans la grille de perception à l'instant t : en effet, détecter toutes les cases traversées par un rayon données n'est pas un problème simple, surtout lorsqu'on s'éloigne du capteur laser. D'un scan laser à l'autre, des "trous" peuvent apparaître dans la grille, correspondant à des cellules contenant une distribution de masses vide ($\mathrm{m}^{\Omega}(\Omega) = 1$). La fusion temporelle permet de remédier à ce problème : d'un instant à l'autre, si une cellule cesse d'être perçue, elle continuera de contenir une distribution de masses "espace libre navigable" (F), pendant le temps nécessaire à l'affaiblissement pour tendre vers l'ignorance.

Finalement, à une distance supérieure à 2m, on note la forte imprécision de replacement des observations vidéos au sein de la grille. Pour illustrer ceci, on montre dans la figure 5.24

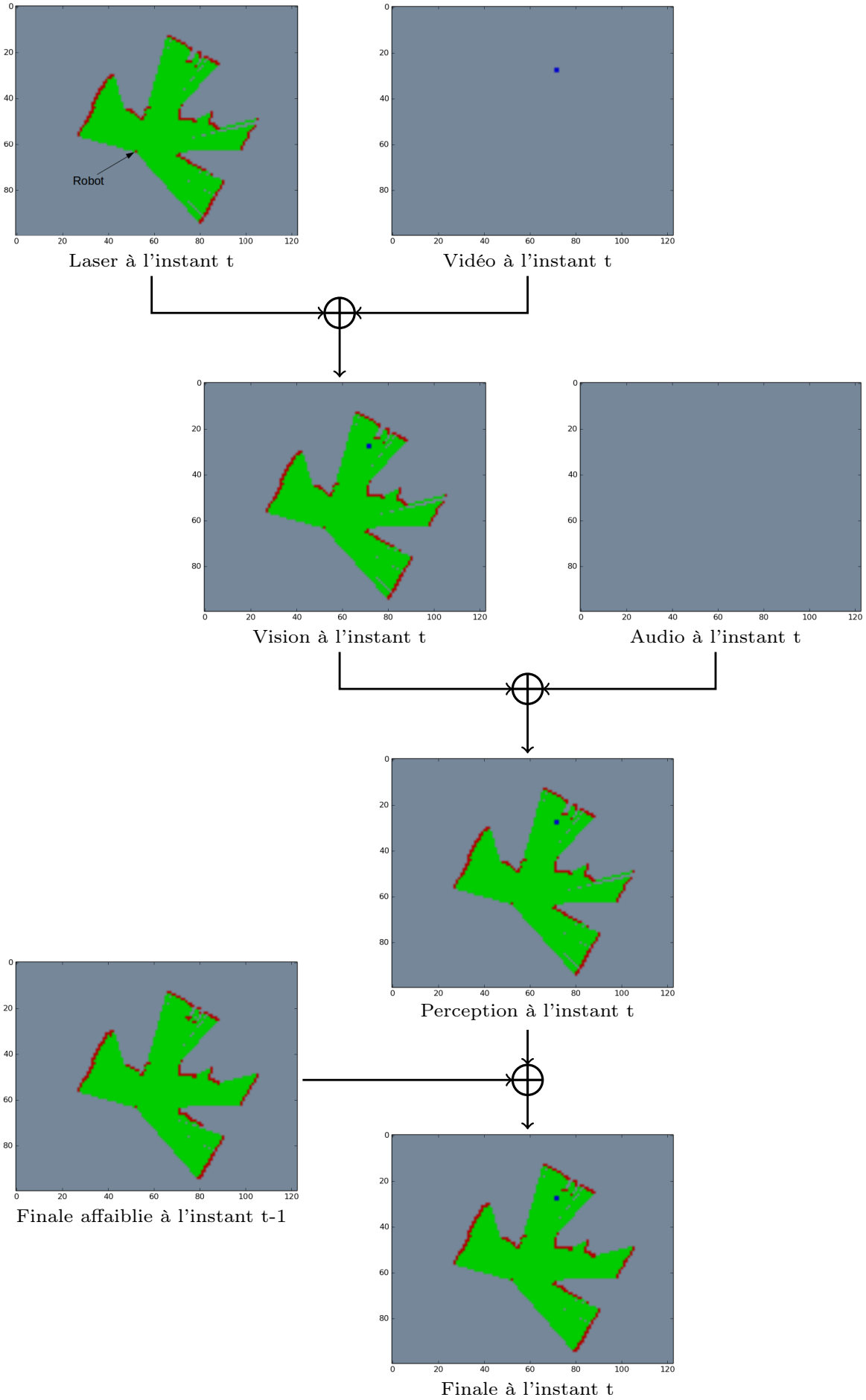
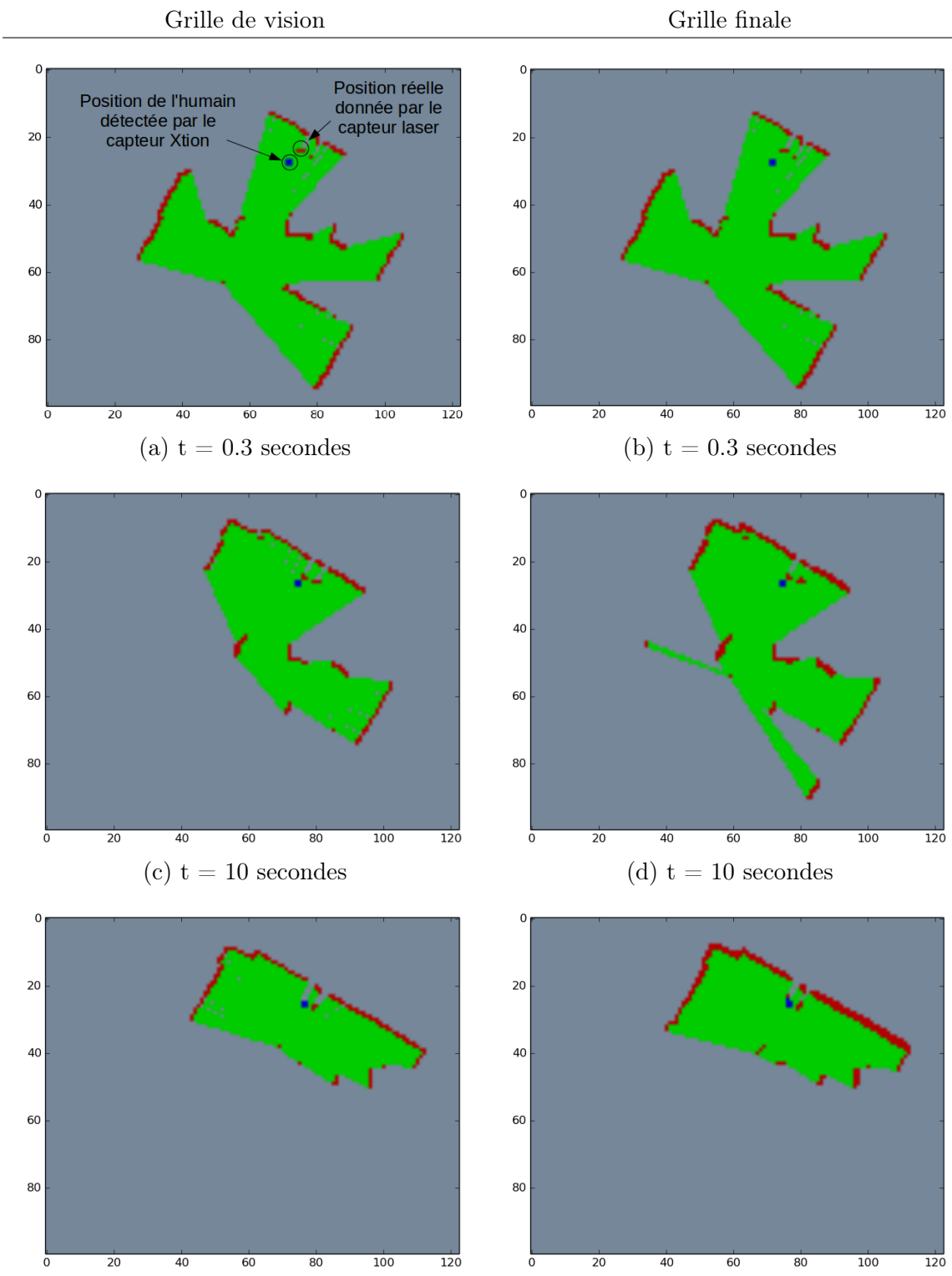


FIGURE 5.23 – Expérience 1 : grilles générées pour le cas d'un humain silencieux

la grille de vision et la grille finale à 3 instants d'observation différents.

La précision du capteur Xtion dépend grandement de la distance de l'humain détecté. Lorsque l'humain détecté se trouve à une distance supérieure à 2m, l'erreur de replacement de l'observation visuelle dans la grille peut atteindre 60cm (figure 5.24 (a)). Lorsque l'humain s'approche et atteint une distance inférieure à 1.50m, les observations vidéos et laser sont superposées sur la grille (figure 5.24 (e)).

Comme on l'a dit, la fusion temporelle n'apporte pas beaucoup d'informations dans ce scénario. Cependant on observe dans la figure 5.24 (f) que les murs détectés par le robot sont plus épais. Ceci provient des erreurs de recalage lors de la localisation du robot : si le robot produit des erreurs lors de l'étape de localisation, la cohérence temporelle entre deux scans laser successifs est perdue. Ce genre d'erreur est plus fréquent lors des rotations du robot.



(a) t = 0.3 secondes (b) t = 0.3 secondes

(c) t = 10 secondes (d) t = 10 secondes

(e) t = 14.5 secondes (f) t = 14.5 secondes

FIGURE 5.24 – Expérience 1 : grille de vision et grille finale lorsque le robot s'approche d'un humain

5.4.2.4 Expérience 2 : translation du robot vers un humain statique silencieux, alors qu'un deuxième humain est source sonore

Dans ce scénario, on souhaite illustrer la fusion audiovisuelle. Cette fois encore, on considère le cas où le robot est en train de se diriger en ligne droite vers un humain précédemment perçu. Cependant on ajoute un deuxième être humain, qui produit du son pendant l'intégralité du déplacement du robot (figure 5.25).

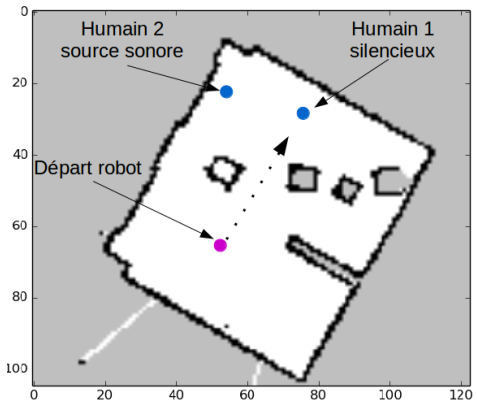


FIGURE 5.25 – Trajectoire suivie par le robot lors du deuxième scénario

A nouveau le processus de construction complet des grilles est illustré par la figure 5.26. La construction de la grille de vision est similaire à celle de l'expérience 1 (figure 5.23). Cependant l'apparition d'un cône audio change la grille de perception à l'instant t : la fusion instantanée donne aux obstacles appartenant au cône une distribution de masses "humain". A nouveau l'impact de la fusion temporelle est ici négligeable sur une itération. On note toutefois que le cône audio dans la grille finale est légèrement plus large que dans la grille de perception. Ceci est dû au mouvement du robot et à l'affaiblissement lors de l'étape de prédiction. Le cône audio persiste dans les zones non perçues par la grille de vision, jusqu'à ce que l'affaiblissement les fasse retourner à une distribution de masses vide.

Trois positions sur la trajectoire du robot sont illustrées par la figure 5.27. Dans cette figure, sur chaque ligne on peut voir 3 grilles à un instant donné : la grille de vision (première colonne), la grille audio (deuxième colonne), et la grille de perception (troisième colonne).

Comme on peut le voir dans la figure (a), à l'instant $t = 7.1$ secondes, l'humain source sonore n'est pas visible par le robot. Le robot étant de petite taille, la table cache l'humain aussi bien au capteur Xtion qu'au capteur laser. Cependant, le cône audio est bien orienté vers la source de son entendue. On peut ainsi voir trois cas de fusion :

— Le cas où une cellule appartient à l'espace libre proche du robot et au cône audio : dans ce cas, comme prévu, c'est au laser que la confiance est accordée, et la croyance est mise en majorité sur l'hypothèse "espace libre navigable" (F).
— Le cas où une cellule appartient au cône audio et à un obstacle détecté par le laser,

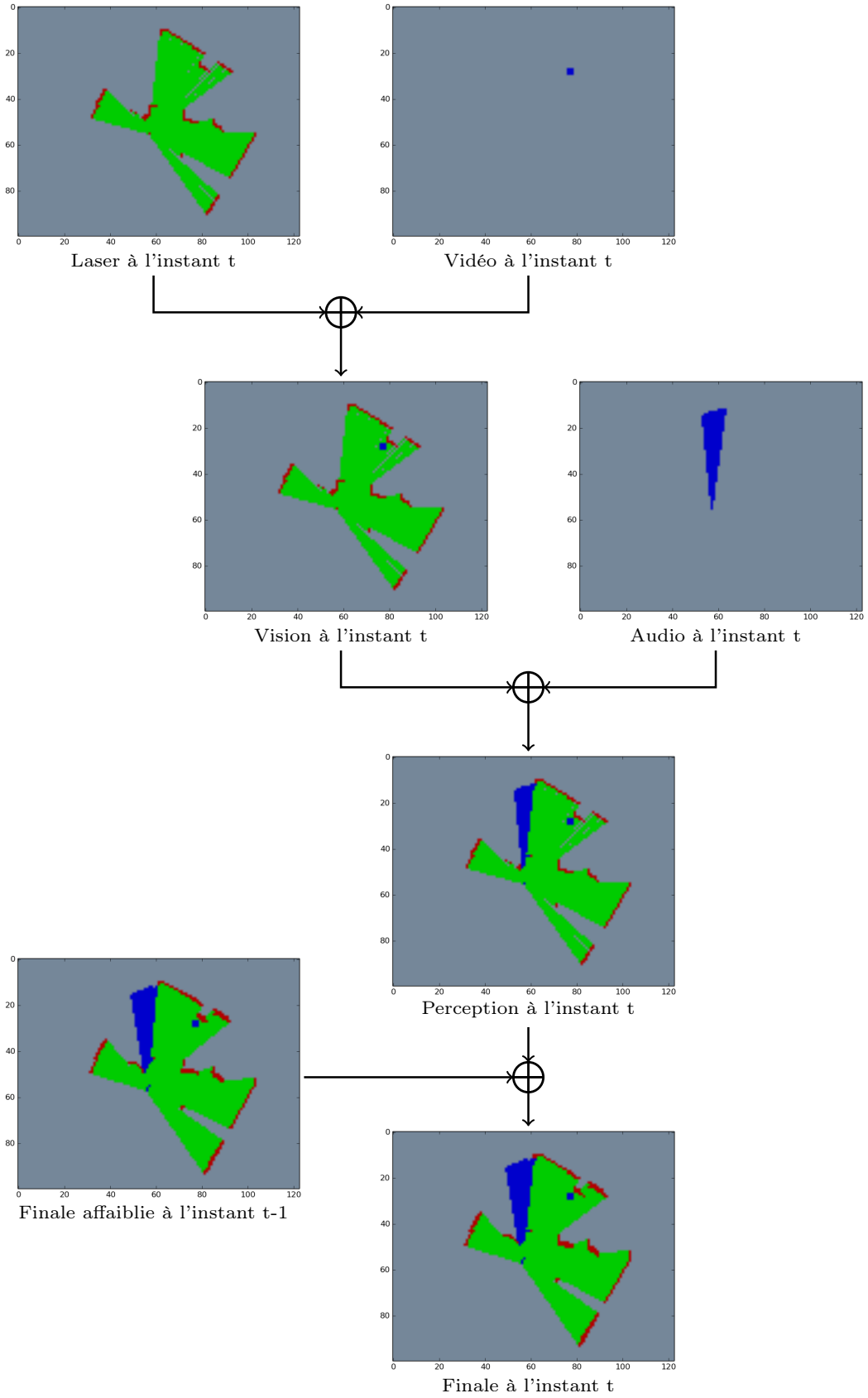


FIGURE 5.26 – Expérience 2 : grilles générées dans le cas d'un humain source sonore

comme c'est le cas pour la table : dans ce cas on fusionne à l'aide d'une fusion conjonctive classique : la croyance majoritaire est donc portée sur l'intersection des hypothèse "Humain ou objet" ($\{H, O\}$, donnée par la grille de vision), et "Humain" (H) (donnée par la grille audio), soit l'hypothèse "Humain" (H). Cela correspond à un principe de prudence.

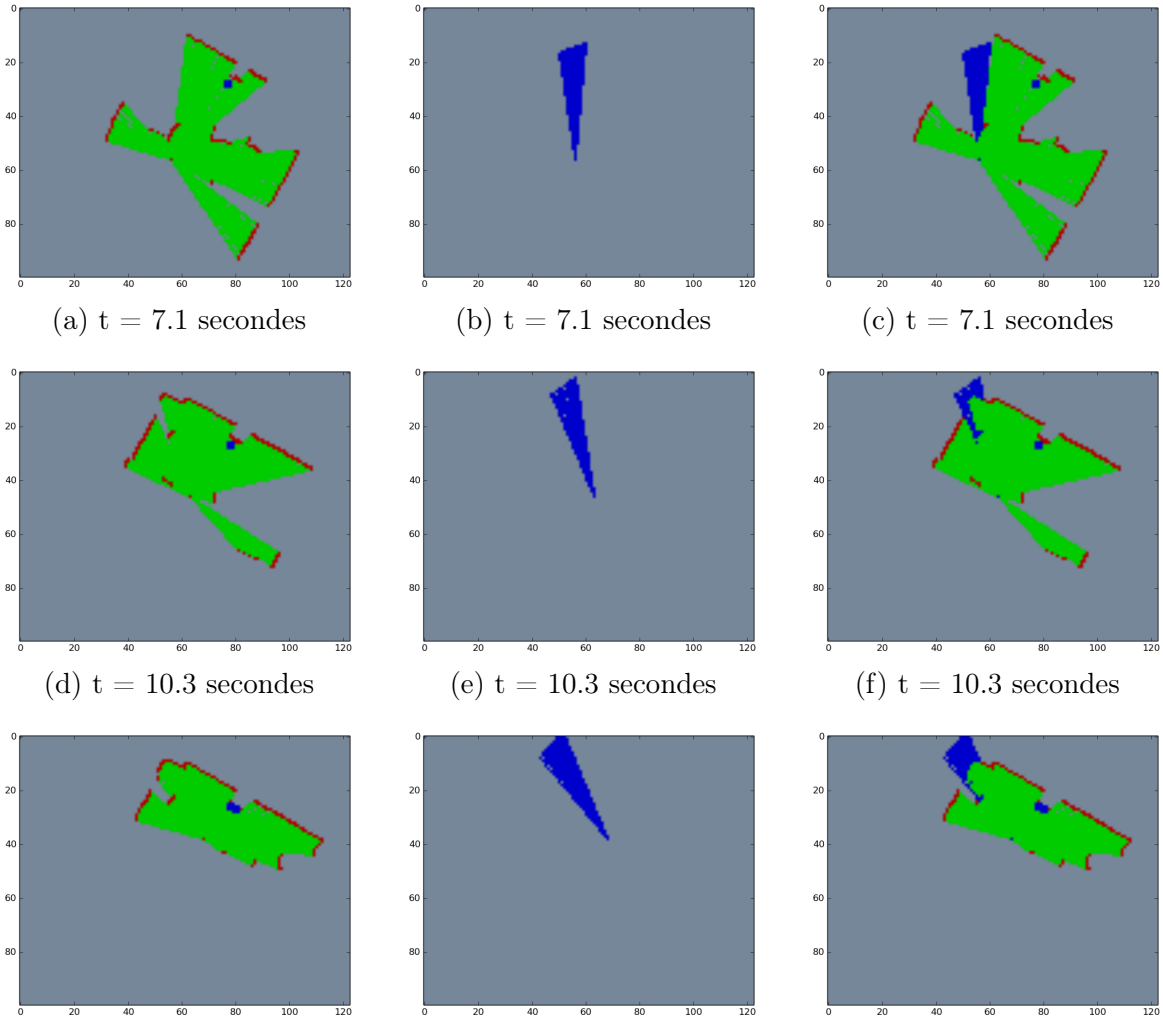
— Le cas où une cellule appartient au cône audio et n'a pas été perçue par la vision (ignorance) : dans ce cas, on retrouve les valeurs exactes du cônes audio, la distribution de masses vide étant l'élément neutre de la fusion conjonctive.

A $t = 10.3$ secondes, l'humain source de son est entré dans le champ de vision du capteur laser. On retrouve le deuxième cas de fusion cité ci-dessus : l'humain est donc bien détecté en tant que tel, sans avoir été détecté par la Xtion. De la même manière, les murs se trouvant dans le cône audio reçoivent une croyance majoritaire sur "H". Comme on peut le voir le cône audio dépasse des murs. Ceci provient de l'implémentation : on limite le cône en portée, mais sans prendre en compte la connaissance a priori du type d'obstacle que peut voir le robot. Il est donc impossible de faire la distinction entre un obstacle statique que le cône peut traverser, ou un mur solide qui devrait stopper le cône audio.

A $t = 12.6$ secondes, le robot a atteint son positionnement par rapport à l'humain détecté visuellement. Le deuxième humain continue à être entendu. Cependant on note que la précision de la détection décroit lorsque la source sonore entendue se trouve sur le côté du robot ($0\,^\circ$ ou $180\,^\circ$). Seule une partie de l'humain fait désormais partie du cône.

Le système réagit donc en accord avec ce qui a été montré simulation, et avec le fonctionnement recherché. La représentation utilisée localise bien l'humain au sein de la grille même lorsque celui-ci n'est pas perçu par le capteur RGB-D.

Vision	Audio	Perception



(a) t = 7.1 secondes (b) t = 7.1 secondes (c) t = 7.1 secondes

(d) t = 10.3 secondes (e) t = 10.3 secondes (f) t = 10.3 secondes

(g) t = 12.6 secondes (h) t = 12.6 secondes (i) t = 12.6 secondes

FIGURE 5.27 – Expérience 2 : grilles de vision, audio, et de perception

5.4.2.5 Performances de l'algorithme

Le capteur laser utilisé fournit des scan laser à une fréquence de 10Hz. C'est la fréquence la plus basse présente au sein du système, et c'est donc cette fréquence que l'on cherche à atteindre pour la création de grilles au cours du temps. 10 grilles par seconde est un compromis acceptable pour détecter précisément les changements au sein de la scène qui entoure le robot. Le tableau 5.3 détaille les performances de l'algorithme de construction de grilles en moyenne sur 1000 constructions de grilles.

Traitement	**Temps Moyen (ms)**
Construction grille laser	**55**
Construction grille vidéo	0.01
Construction grille de vision	1
Construction grille grille audio	2
Fusion pour grille de perception	**30**
Fusion temporelle pour grille finale	**28**
Prédiction du HMM	2
Total	118ms (8.5Hz)

TABLE 5.3 – Performances de l'algorithme de création de grilles sur l'espace de discernement $\Omega = \{H, O, F\}$

L'opération la plus coûteuse en temps de calcul est le lancé de rayon lors de la construction de la grille laser et prend plus de la moitié du temps dont on dispose. Ce coût est irréductible sans perdre une partie de l'information (par exemple ne traiter qu'un rayon sur 2 permettrait de diviser le temps de construction de la grille laser par 2).

Sans surprise, les deux autres opérations les plus coûteuses sont les fusions : une fusion instantanée pour la construction de grille de perception, et une fusion temporelle pour la construction de la grille finale. Il est important de noter que ces temps de construction ne dépendent que peu de la taille de la grille mise à jour. Elle dépend principalement de l'aire que peut percevoir le robot à un instant donné : plus cette aire est grande, plus le coût de calcul sera élevé.
On arrive toutefois à atteindre dans cette situation une fréquence de construction de grilles de 8.5Hz.

Pour diminuer le temps de calcul, on peut stocker les différentes masses possibles résultantes de la fusion instantanée en mémoire du robot pour gagner du temps. Ceci permet de pré-calculer la fusion avant la construction des grilles en temps réel : comme on connaît l'intégralité des distributions de masses existantes dans les modèles capteurs inverses, il existe un nombre fini de combinaisons possibles. Dans cette situation, le tableau 5.4 montre qu'on atteint le temps capteur recherché, avec une chute drastique du coût de la fusion pour la grille de perception.

Traitement	Temps Moyen (ms)
Construction grille laser	**55**
Construction grille vidéo	0.01
Construction grille de vision	1
Construction grille grille audio	2
Fusion pour grille de perception	2
Fusion temporelle pour grille finale	**28**
Prédiction du HMM	2
Total	90 ms (11 Hz)

TABLE 5.4 – Performances de l'algorithme de création de grilles sur l'espace de discernement $\Omega = \{H, O, F\}$, avec pré-calcul de la fusion instantanée

5.5 Conclusion

Un système de cartographie évidentielle a été présenté au sein de ce chapitre. Il repose sur une fusion multimodale, intégrant les données de différents capteurs : un capteur laser, un capteur RGB-D, et une paire de microphones. 2 fusions successives ont lieu : une fusion instantanée donnant accès à une grille de perception, concentrant l'intégralité de l'information perçue à un instant donné par le robot. Cette grille est par la suite intégrée dans une fusion temporelle prenant en compte les informations perçues aux instants précédents. Ces grilles incluent l'être humain directement au sein des cartes et présentent plusieurs avantages :

— Elles sont adaptables aux types de capteurs employés : les opérateurs de fusion de fonctions de croyance permettent une grande flexibilité du transfert de la croyance. Par exemple, l'imprécision de la localisation de source sonore peut être compensée par un contrôle du conflit lors de la construction de la grille de perception,

— Elles modélisent le mouvement dans l'environnement du robot. L'introduction de l'espace de discernement $\Theta = \{H_s, H_m, O_s, O_m\}$ permet une description précise de la dynamique de l'environnement du robot,

— Elle peuvent permettre à un robot compagnon de prendre des décisions directement par rapport aux humains de son entourage, et de planifier ses trajectoires en les prenant en compte (par exemple, un robot peut choisir d'éviter au maximum les endroits où un humain mobile a été détecté, pour ne pas gêner le déplacement des humains qui l'entourent).

Le système a été testé intégralement en simulation : les deux espaces de discernement Ω et Θ ont pu être comparés sur le même scénario et l'enrichissement apporté par Θ a été montré. En implémentation réelle, l'espace de discernement Ω a été testé sur le robot au sein de deux expériences et donne des résultats représentant correctement la réalité de la scène qui entoure le robot en temps réel en déportant les calculs sur ordinateur annexe.

Chapitre 6

Conclusion générale et perspectives

Récapitulation des contributions

La thématique des robots compagnons couvre un vaste champ de recherche. L'objectif fixé est de rendre ces robots capables de détecter et localiser des humains dans leur environnement dans le but ultérieur d'analyser leur comportement. Le robot est ainsi vu comme un système de perception active.
Cette thèse a permis de proposer quelques contributions liées à la perception de l'environnement par ces robots destinés à partager la vie d'êtres humains. Pour cela, différentes fonctionnalités qui ont vocation à être intégrées dans un système robotique ont été développées :

Classification d'événements sonores en environnement intérieur : la méthode de classification présentée ici, destinée à la détection de marqueurs de présence humaine, repose sur un petit nombre de classes : "Parole", "Musique", "Impact", "Son de l'environnement". L'utilisation d'une taxonomie de petite taille permet de collecter les informations importantes relatives à la mission du robot (présence d'humains), tout en évitant une trop grande spécificité, qui pourrait poser problème en cas d'erreur de classification. Un autre atout de ce système de classification est l'utilisation des fonctions de croyance pour la classification. L'audio étant une modalité aisément soumise au bruit, il est important d'inclure l'incertitude dans la classification utilisée, qui permet de reconnaître certains sons comme étant "inconnus". La méthode a été testée sur une base de sons issue de différentes bases de sons existantes.

Fusion audiovisuelle pour la détection de locuteurs successifs dans une conversation : ce système permet au robot de repérer la personne en train de parler à chaque instant d'observation dans la vidéo. Il repose sur une fusion audiovisuelle probabiliste et fonctionne à la cadence capteur sur une caméra de faible résolution. De plus, le système s'adapte automatiquement à la scène à laquelle le robot assiste à l'aide d'un apprentissage en ligne de la couleur de peau des personnes présentes. La méthode a été testée sur des vidéos du robot et fonctionne en situation réelle, avec des personnes mobiles ou non, présentes dans le champ de vision du robot ou non.

Navigation dédiée à la détection d'humains à l'aide d'une fusion multimodale : ce travail permet à un robot mobile de se déplacer de manière autonome dans son environnement à la recherche d'êtres humains. Il repose sur la construction d'une grille métrique de perception multimodale regroupant l'intégralité de l'information perçue à chaque instant d'observation. Cette grille améliore la capacité de représentation de l'environnement du robot

apportée par les grilles d'occupation classiquement construites dans la littérature, et permet au robot de prendre une décision sur l'information prioritaire. La décision quant au prochain objectif du robot est prise à l'aide d'un automate et la planification de trajectoire est faite grâce à un diagramme de Voronoï extrait d'une grille d'occupation construite hors-ligne. La grille de perception proposée est adaptée à des capteurs aux données hétérogènes, et utilise des modèles capteurs inverses simples qui représentent efficacement les données. Il a été implémenté et testé en conditions réelles sur un robot mobile.

Modélisation crédibiliste de l'environnement pour la navigation : ce système améliore le processus de construction de la grille de perception multimodale décrite ci-dessus en utilisant le formalisme des fonctions de croyance. Ceci permet la création de modèles capteurs inverses prenant en compte l'incertitude liée aux données. De plus, les fonctions de croyance disposent d'opérateurs de fusion qui permettent de prendre facilement en compte les cas difficiles de fusion, à l'aide d'une gestion du conflit capteur généré dans le temps. Ce conflit est utilisé pour décrire explicitement la dynamique de la scène. La représentation proposée est performante et améliore le potentiel de prise de décision du robot. Cette méthode de cartographie a été implémentée et testée dans des conditions réelles sur le robot.

Mise en place des algorithmes proposés sur une plateforme matérielle : les contributions diverses de cette thèse répondent toutes à une problématique scientifique particulière, qui s'inscrit dans le cadre de la robotique de services. Il est intéressant de noter que les deux dernières contributions ont été implémentées sur un robot mobile, ce qui apporte des challenges techniques : le robot dispose de ressources de calculs limitées, ce qui rend le traitement à cadence capteur difficile. Une solution de déportation de calcul sur un ordinateur tiers a été adoptée ici, mais elle est sensible à la condition du réseau dans lequel se trouve le robot.
De plus, le travail sur une plateforme matérielle rend le système dépendant des capteurs employés, ainsi que de la qualité de l'information fournie par ces capteurs.

Perspectives

Perspectives à moyen-terme

Implémentation du système de classification d'événements sonores directement sur le robot : adapter cette classification à un flux audio permettrait au robot d'augmenter grandement sa capacité de compréhension de la scène qui l'entoure. Cependant, la transition vers un flux audio, avec une classification "au fil de l'eau", n'est pas directe : elle nécessite l'ajout d'un modèle de segmentation d'évènements sonores à différentes échelles temporelles, qui permettrait au robot de séparer les différents sons dans le temps. Il existe de tels dispositifs dans la littérature [Vac+14], qu'il est possible d'adapter à notre cas.

Par ailleurs, il est important d'adapter la base d'apprentissage à l'environnement du robot. De manière générale, la base d'apprentissage est obtenue à l'aide d'un système expert (humain

ou non), donnant accès à des données étiquetées. Un tel genre de système n'est pas disponible à l'heure actuelle pour un flux audio. Pour répondre à ce problème, il serait possible d'utiliser des méthodes d'apprentissage par renforcement : le robot devrait dans ce cas classifier de façon non-supervisée une base de sons entendus dans des classes sans étiquette. Une fois les classes apprises, un expert devrait vérifier si ces classes ont du sens pour la mission du robot. Si c'est le cas, l'apprentissage peut être utilisé tel quel. Dans le cas contraire, l'apprentissage devrait être ajusté de manière guidée par l'humain pour obtenir des classes sémantiquement convaincantes.

Amélioration de la fusion de grilles évidentielles pour prédire la trajectoire d'objets : Dans le chapitre 5, lors de la fusion à l'aide du HMM crédibiliste, on fusionne les différentes grilles cellule par cellule, sans prendre en compte l'information des voisines. La gestion du conflit permet de déterminer si la cellule vient de se remplir (auquel cas on considère que quelque chose ou quelqu'un de mobile vient d'entrer dans cette cellule). Cependant, l'information de trajectoire des objets mobiles détectés ne peut pas être obtenue avec le mécanisme utilisé actuellement. Il serait intéressant de prendre en compte le voisinage de chaque cellule lors de l'étape de prédiction du HMM : une fonction de voisinage permettrait de prévoir le déplacement d'un objet ou d'un humain mobile, donc sa trajectoire. Cependant, la construction d'une telle étape de prédiction n'est pas facile : elle nécessite la création d'une matrice de transition de dimension dépendant du voisinage observé, et augmenterait sensiblement le coût de calcul.

Perspectives à long-terme

A plus long terme, il existe d'autres axes d'amélioration des travaux effectués au cours de cette thèse. Ils concernent principalement l'utilisation du cadre crédibiliste. Les fonctions de croyance semblent particulièrement adaptées à la robotique. En effet, dans la perspective d'élaborer une intelligence artificielle destinée à vivre avec les êtres humains, la capacité de représentation du robot est importante. Or, le cadre crédibiliste donne une grande flexibilité de représentation du savoir, qui inclut l'ignorance. Cette représentation explicite du doute a un fort potentiel pour les robots compagnons équipés de plusieurs types de capteurs.

Représentation hiérarchique de l'environnement du robot : dans le chapitre 5, les informations qui sont fusionnées au sein de grilles évidentielles sont de natures similaires : elles représentaient un objet physique (humain, obstacle, espace libre). Toutefois différents capteurs peuvent donner des informations hétérogènes. Par exemple, dans le cas où la classification d'événements sonores serait mise en place sur le robot, celui-ci devrait inclure des concepts de plus bas niveau sémantique dans sa représentation de l'environnement : un impact entendu peut avoir une origine humaine, mais il peut provenir d'un objet tombé à terre ou d'une fenêtre qui claque. Par ailleurs, la connaissance préalable des objets statiques de l'environnement par le robot peut lui donner la possibilité d'associer des données de natures différentes (par exemple, si un son est entendu à un endroit où le robot sait qu'il y a un poste de télévision, cela devrait pouvoir l'encourager à aller observer les chaises se trouvant en face du poste). Pour cela, l'utilisation d'ontologies et la hiérarchisation des données dans une fusion évidentielle semble

adaptée. Certains travaux allant dans ce sens ont été effectués : [Hon+09] propose une fusion évidentielle faisant le lien entre les données de différents capteurs (ouverture de la porte du réfrigérateur) et des concepts de haut niveau (se préparer une boisson froide), au sein d'une maison intelligente.

Prise de décision sur les grilles évidentielles : L'augmentation de la capacité de compréhension d'un robot doit s'accompagner de méthodes pour prendre des décisions quant à ses prochains déplacements en prenant en compte l'information perçue. Il serait intéressant d'explorer les possibilités de prise de décisions sur des grilles évidentielles. A ce jour, il n'existe que très peu de travaux portant sur ce sujet [CRK16]. L'idéal serait de permettre au robot de trouver facilement l'information utile pour remplir la mission qui lui a été fixée dans l'espace de représentation dont il dispose. La hiérarchisation des données décrite ci-dessus peut être utile pour une telle mission : à partir de données capteurs hétérogènes et de grande dimension, obtenir un espace de discernement de petite taille contenant les concepts sémantiques importants pour le robot peut l'aider à définir efficacement ses objectifs.

Liste des publications

R. Ratajczak, D. Pellerin, **Q. Labourey**, et C. Garbay, "A Fast Audiovisual Attention Model for Human Detection and Localization on a Companion Robot", International Conference on Applications and Systems of Visual Paradigms (VISUAL), Barcelone, 2016

Q. Labourey, O. Aycard, D. Pellerin, M. Rombaut et C. Garbay, "An Evidential Filter for Indoor Navigation of a Mobile Robot in Dynamic Environment", International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU), Eindhoven, 2016

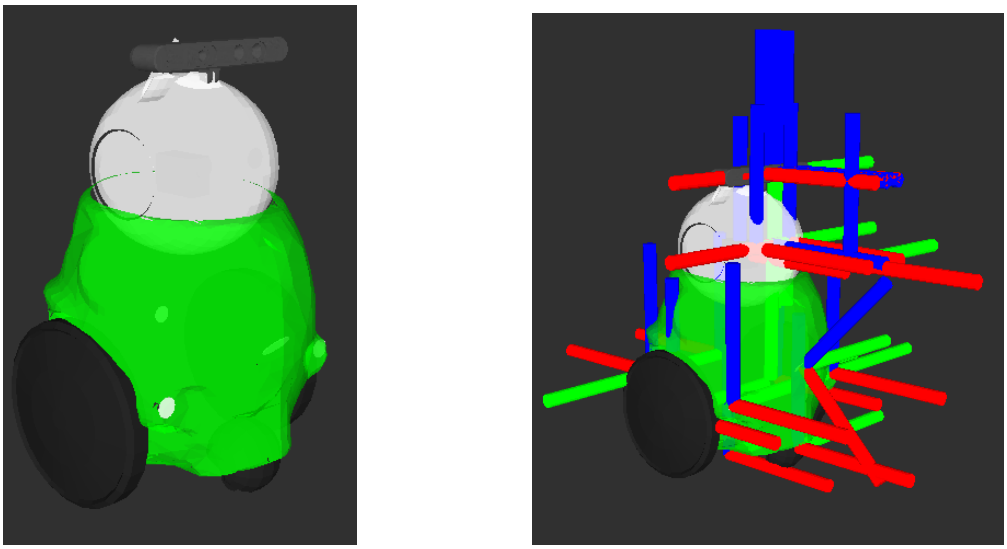
Q. Labourey, D. Pellerin, M. Rombaut, O. Aycard et C. Garbay, "Sound classification in indoor environment thanks to belief functions", European Signal Processing Conference (EUSIPCO), Nice, 2015

Q. Labourey, O. Aycard, D. Pellerin et M. Rombaut, "Audiovisual data fusion for successive speakers tracking", International Conference on Computer Vision Theory and Applications (VISAPP), Lisbonne, 2014

Annexe A

Arbre cinématique complet du robot

L'arbre cinématique simplifié du robot a été présenté dans le chapitre 4. Toutefois, nous avons construit un arbre intégral, incluant d'autres capteurs non évoqués au sein de ce manuscrit : le robot dispose par exemple de deux capteurs à ultra sons, sur chaque côté. Par ailleurs, chaque caméra du capteur RGB-D reçoit deux systèmes de coordonnées, l'un correspondant au repère "image" et l'autre correspondant au repère capteur. L'ensemble des différents systèmes de coordonnées est représenté dans la figure A.1.



(a) Robot représenté en 3D

(b) Robot et l'arbre cinématique complet

Figure A.1 – Ensemble des systèmes de coordonnées qui composent le robot

Ainsi pour transformer la position d'une observation obtenue dans un repère donné, il est nécessaire de connaître l'arbre cinématique, qui décrit les transformées entre les différents repères qui composent le robot. Cet arbre est décrit au sein d'un fichier de format Unified Robot Description Format. Il définit l'emplacement dans l'espace des différents repères qui composent le robot. Chaque repère possède un repère parent, et peut avoir des repères enfants. L'arbre cinématique intégral du robot Q.bo est illustré par la figure A.2

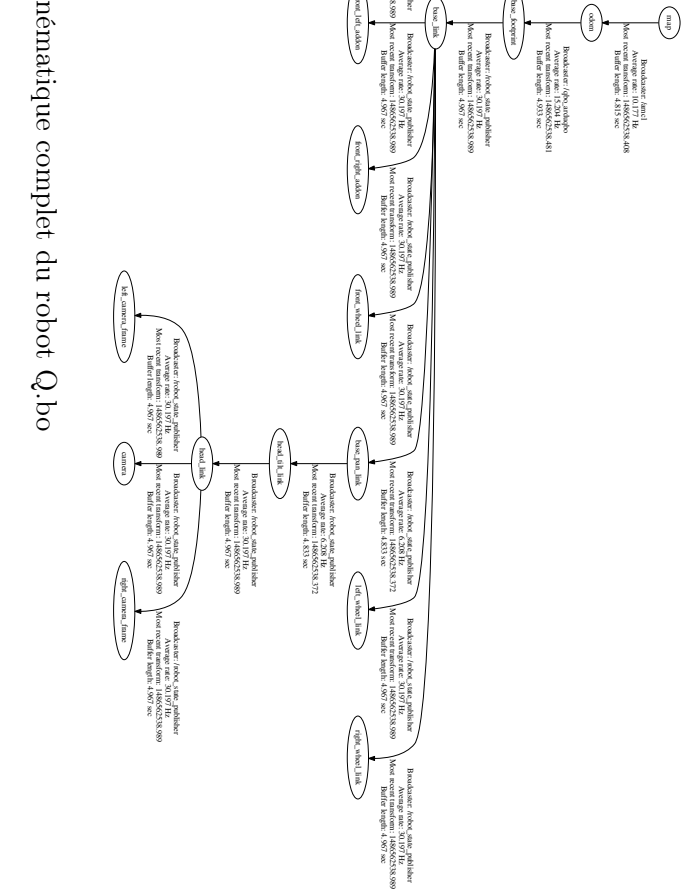


FIGURE A.2 – Arbre cinématique complet du robot Q.bo

Annexe B

Architecture logicielle intégrale

On souhaite illustrer l'architecture logicielle intégrale utilisée lors de la navigation autonome du robot. L'ensemble des noeuds ROS actifs, ainsi que les différents topics lorsque le robot est en cours de navigation sont exposés dans la figure B.1. Ces noeuds représentent plusieurs milliers de lignes de code réparties sur l'ensemble de l'architecture, en deux langages de programmation différents : le C++ et le Python.

Pour faire plus facilement la correspondance avec la figure 4.25, où les noms des noeuds présentés ont été changés pour faciliter la compréhension, on propose un tableau des correspondances (tableau B.1).

Architecture simplifiée (figure 4.25)	Architecture intégrale (figure B.1)
qbo_arduino	qbo_arduqbo
qbo_description	Non représenté : l'arbre cinématique n'est lu qu'une fois
hokuyo_node	/hokyuo
openni2	intégralité des noeuds commençant par /camera/...
qbo_audiocapture	/audio_capture
qbo_cinematique	/robot_state_publisher
qbo_detectionhumains	/colordetector
qbo_replacement	/qbo_transform_facePos
qbo_localisationsonore	/qbo_soundlocalization
amcl	amcl
qbo_perceptionmultimodale	/Itinerary_organizer
gmapping	Non représenté : la grille d'occupation est donnée par /map_server
move_base	move_base : qbo_voronoi et local_planner ne sont pas représentés car directement inclus dans move_base

Table B.1 – Correspondance entre les noms de noeuds figurant dans la figure 4.25 et la figure B.1

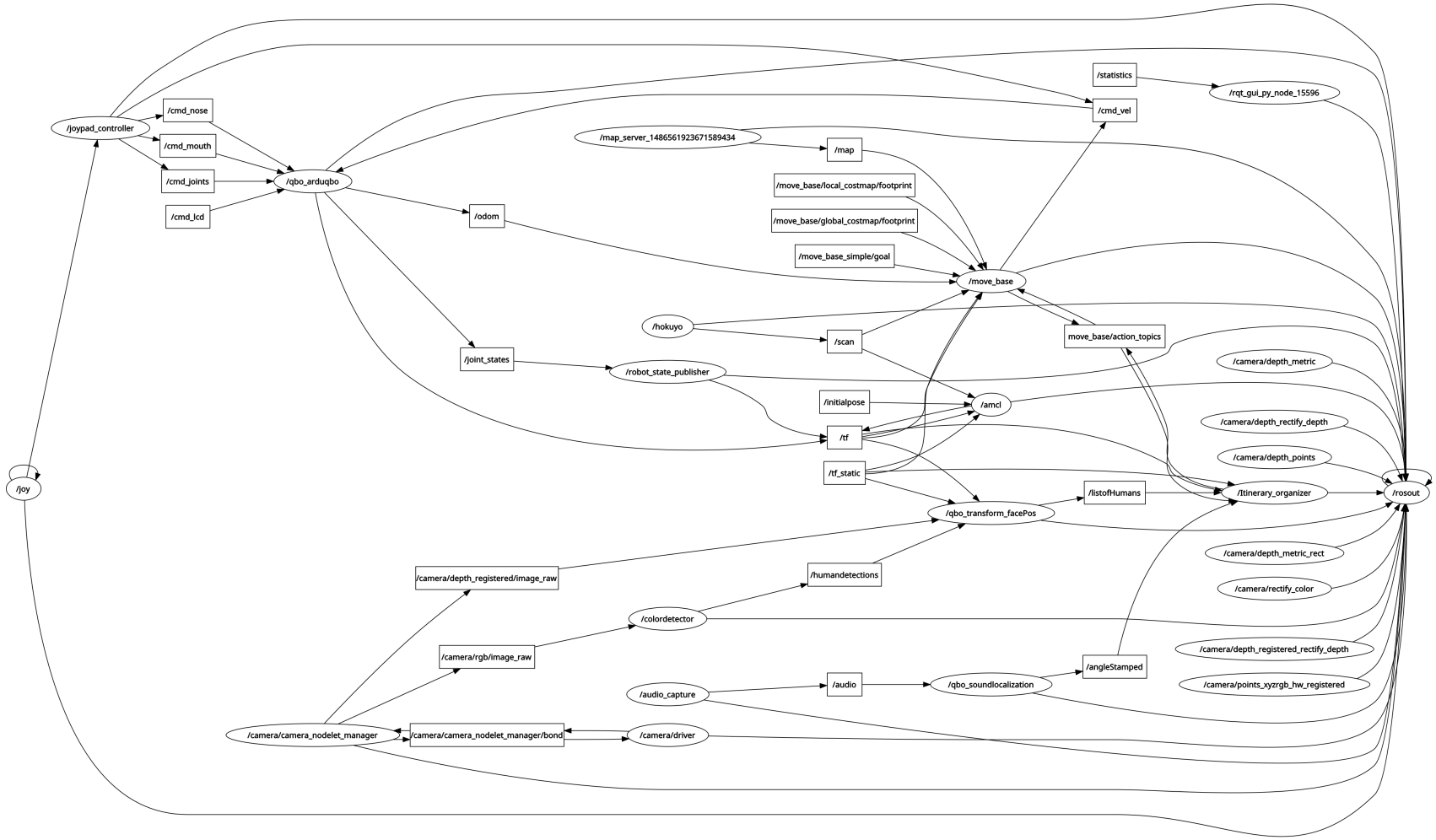


FIGURE B.1 – Architecture logicielle intégrale : les noeuds ROS sont représentés sous forme d'ellipses et les topics sous forme de boîtes.

Bibliographie

[Ada+03] W. H. Adams, G. Iyengar, C. Lin, M. R. Naphade, C. Neti, H. J. Nock et J. R. Smith. "Semantic Indexing of Multimedia Content Using Visual, Audio, and Text Cues". In : *EURASIP Journal on Advances in Signal Processing* (2003) (cf. p. 49).

[AKJ06] P. K. Atrey, M. S. Kankanhalli et R. Jain. "Information assimilation framework for event detection in multimedia surveillance systems". In : *Multimedia Systems* (2006) (cf. p. 46).

[AM79] B. D. Anderson et J. B. Moore. *Optimal filtering*. Courier Corporation, 1979 (cf. p. 86).

[AR76] B. Atal et L. Rabiner. "A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition". In : *IEEE Transactions on Acoustics, Speech, and Signal Processing* (1976) (cf. p. 32).

[Aru+02] M. S. Arulampalam, S. Maskell, N. Gordon et T. Clapp. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking". In : *IEEE Transactions on Signal Processing* (2002) (cf. p. 48, 83, 86).

[Atr+10] P. K. Atrey, M. A. Hossain, A. El Saddik et M. S. Kankanhalli. "Multimodal fusion for multimedia analysis : a survey". In : *Multimedia Systems* (2010) (cf. p. 43, 48).

[Ayc10] O. Aycard. "Contribution to Perception for Intelligent Vehicles". Accreditation to supervise research. Université de Grenoble, 2010 (cf. p. 81).

[BC07] H. Bredin et G. Chollet. "Audio-Visual Speech Synchrony Measure for Talking-Face Identity Verification". In : *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2007 (cf. p. 49).

[BHR09] J. Broekens, M. Heerink et H. Rosendal. "Assistive social robots in elderly care : a review". In : *Gerontechnology* (2009) (cf. p. 7).

[BL08] A. Baggio et K. Langendoen. "Monte Carlo localization for mobile wireless sensor networks". In : *Ad Hoc Networks* (2008) (cf. p. 86).

[CH67] T. Cover et P. Hart. "Nearest neighbor pattern classification". In : *IEEE Transactions on Information Theory* (1967) (cf. p. 16, 27).

[Cha+12] I. Chaari, A. Koubaa, H. Bennaceur, S. Trigui et K. Al-Shalfan. "smartPATH : A hybrid ACO-GA algorithm for robot path planning". In : *Evolutionary Computation (CEC), 2012 IEEE Congress on*. 2012 (cf. p. 88).

[CHMM12] L. Chen, H. Hu et K. McDonald-Maier. "EKF Based Mobile Robot Localization". In : *International Conference on Emerging Security Technologies (EST)*. 2012 (cf. p. 86).

[CRK16] J. Clemens, T. Reineking et T. Kluth. "An evidential approach to SLAM, path planning, and active exploration". In : *International Journal of Approximate Reasoning* (2016) (cf. p. 132–134, 136, 174).

[Dau+05] K. Dautenhahn, S. Woods, C. Kaouri, M. L. Walters, K. L. Koay et I. Werry. "What is a robot companion - friend, assistant or butler ?" In : *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005 (cf. p. 5).

[Dem08] A. P. Dempster. In : *Classic Works of the Dempster-Shafer Theory of Belief Functions*. 2008. Chap. A Generalization of Bayesian Inference (cf. p. 25).

[Den08] T. Denœux. "Conjunctive and disjunctive combination of belief functions induced by nondistinct bodies of evidence". In : *Artificial Intelligence* (2008) (cf. p. 25).

[Den95] T. Denoeux. "A k-nearest neighbor classification rule based on Dempster-Shafer theory". In : *IEEE Transactions on Systems, Man and Cybernetics* (1995) (cf. p. 28, 29, 31, 37).

[Dij59] E. W. Dijkstra. "A note on two problems in connexion with graphs". In : *Numerische Mathematik* (1959) (cf. p. 88).

[Dis+11] G. Dissanayake, S. Huang, Z. Wang et R. Ranasinghe. "A review of recent developments in simultaneous localization and mapping". In : *IEEE International Conference on Industrial and Information Systems*. 2011 (cf. p. 81).

[Dun73] J. C Dunn. "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters". In : *Journal of Cybernetics* (1973) (cf. p. 16).

[Elf89] A. Elfes. "Using occupancy grids for mobile robot perception and navigation". In : *Computer* (1989) (cf. p. 81).

[FBT97] D. Fox, W. Burgard et S. Thrun. "The dynamic window approach to collision avoidance". In : *IEEE Robotics Automation Magazine* (1997) (cf. p. 90, 105).

[FJL12] M. F. Fallon, H. Johannsson et J. J. Leonard. "Efficient scene simulation for robust monte carlo localization using an RGB-D camera". In : *IEEE International Conference on Robotics and Automation (ICRA)*. 2012 (cf. p. 86).

[Foo97] J. T. Foote. "Content-based retrieval of music and audio". In : *Voice, Video, and Data Communications*. 1997 (cf. p. 17).

[Fox+99] D. Fox, W. Burgard, F. Dellaert et S. Thrun. "Monte carlo localization : Efficient position estimation for mobile robots". In : *AAAI/IAAI* (1999) (cf. p. 86).

[GD15] D. Gamage et T. Drummond. "Reduced dimensionality extended Kalman Filter for SLAM in a relative formulation". In : *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015 (cf. p. 85).

[GSB07] G. Grisetti, C. Stachniss et W. Burgard. "Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters". In : *IEEE Transactions on Robotics* (2007) (cf. p. 84, 97, 112, 133).

[Hah+03] D. Hahnel, W. Burgard, D. Fox et S. Thrun. "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements". In : *IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*. IEEE. 2003 (cf. p. 84).

[HB05] W. H. Huang et K. R. Beevers. "Topological Mapping with Sensing-Limited Robots". In : *Algorithmic Foundations of Robotics VI*. 2005 (cf. p. 80, 81, 89).

[HK13] M. A. Haque et J. Kim. "An analysis of content-based classification of audio signals using a fuzzy c-means algorithm". In : *Multimedia Tools and Applications* (2013) (cf. p. 18, 34).

[HL97] D. L. Hall et J. Llinas. "An introduction to multisensor data fusion". In : *Proceedings of the IEEE* (1997) (cf. p. 43).

[HNR68] P. E. Hart, N. J. Nilsson et B. Raphael. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". In : *IEEE Transactions on Systems Science and Cybernetics* (1968) (cf. p. 88).

[Hon+09] X. Hong, C. Nugent, M. Mulvenna, S. McClean, B. Scotney et S. Devlin. "Evidential fusion of sensor data for activity recognition in smart homes". In : *Pervasive and Mobile Computing* (2009) (cf. p. 51, 174).

[HS10] O. E. Hamzaoui et B. Steux. "A fast scan matching for grid-based laser SLAM using streaming SIMD extensions". In : *International Conference on Control Automation Robotics Vision (ICARCV)*. 2010 (cf. p. 84).

[Ifr] *Fédération Internationale de la robotique*. `http://www.ifr.org/` (cf. p. 4, 5).

[INN03] G. Iyengar, H. J. Nock et C. Neti. "Audio-visual synchrony for detection of monologues in video archives". In : *International Conference on Acoustics, Speech, and Signal Processing*. 2003 (cf. p. 48).

[Ins] *INSEE : Projection de population pour la France métropolitaine à l'horizon 2050* (cf. p. 7).

[IY08] A. Itai et H. Yasukawa. "Footstep classification using simple speech recognition technique". In : *IEEE International Symposium on Circuits and Systems*. 2008 (cf. p. 19, 34).

[Jan+12] M. Janvier, X. Alameda-Pineda, L. Girin et R. Horaud. "Sound-Event Recognition with a Companion Humanoid". In : *IEEE International Conference on Humanoid Robotics*. 2012 (cf. p. 19–21).

[JL05] I. Jung et S. Lacroix. "Simultaneous Localization and Mapping with Stereovision". In : *Robotics Research*. 2005 (cf. p. 85).

[Kav+96] L. E. Kavraki, P. Svestka, J. C. Latombe et M. H. Overmars. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In : *IEEE Transactions on Robotics and Automation* (1996) (cf. p. 81, 88, 89).

[Kha+13] B. Khaleghi, A. Khamis, F. O. Karray et S. N. Razavi. "Multisensor data fusion : A review of the state-of-the-art". In : *Information Fusion* (2013) (cf. p. 48).

[Kha85] O. Khatib. "Real-time obstacle avoidance for manipulators and mobile robots". In : *IEEE International Conference on Robotics and Automation (ICRA)*. 1985 (cf. p. 90).

[KS06] D. H. Kim et S. Shin. "Local path planning using a new artificial potential function composition and its analytical design guidelines". In : *Advanced Robotics* (2006) (cf. p. 90).

[Kui78] B. Kuipers. "Modeling spatial knowledge". In : *Cognitive science* (1978) (cf. p. 79).

[Kun+06] V. Kunchev, L. Jain, V. Ivancevic et A. Finn. "Path Planning and Obstacle Avoidance for Autonomous Mobile Robots : A Review". In : *Knowledge-Based Intelligent Information and Engineering Systems : 10th International Conference*. 2006 (cf. p. 90).

[Kur+12] M. Kurdej, J. Moras, V. Cherfaoui et P. Bonnifait. "Map-Aided Fusion Using Evidential Grids for Mobile Perception in Urban Environment". In : *Belief Functions : Theory and Applications*. 2012 (cf. p. 85).

[Kur+14] M. Kurdej, J. Moras, V. Cherfaoui et P. Bonnifait. "Controlling remanence in evidential grids using geodata for dynamic scene perception". In : *International Journal of Approximate Reasoning* (2014) (cf. p. 131).

[Kur+15] M. Kurdej, J. Moras, V. Cherfaoui et P. Bonnifait. "Map-aided evidential grids for driving scene understanding". In : *IEEE Intelligent Transportation Systems Magazine* (2015) (cf. p. 51, 131, 134, 135, 147).

[Lab+14] Q. Labourey, O. Aycard, D. Pellerin et M. Rombaut. "Audiovisual data fusion for successive speakers tracking". In : *International Conference on Computer Vision Theory and Applications*. 2014 (cf. p. 43).

[Lab+15] Q. Labourey, D. Pellerin, M. Rombaut, O. Aycard et C. Garbay. "Sound classification in indoor environment thanks to belief functions". In : *23rd European Signal Processing Conference (EUSIPCO)*. 2015 (cf. p. 14).

[Lab+16] Q. Labourey, O. Aycard, D. Pellerin, M. Rombaut et C. Garbay. "An Evidential Filter for Indoor Navigation of a Mobile Robot in Dynamic Environment". In : *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*. 2016 (cf. p. 51, 126).

[LGG04] A. P. Loh, F. Guan et S. S. Ge. "Motion estimation using audio and video fusion". In : *Control, Automation, Robotics and Vision Conference*. 2004 (cf. p. 48).

[LHL09] X. Li, J. He et X. Liu. "Ant Intelligence for Solving Optimal Path-covering Problems with Multi-objectives". In : *International Journal of Geographical Information Science* (2009) (cf. p. 88).

[Li00] S. Z. Li. "Content-based audio classification and retrieval using the nearest feature line method". In : *IEEE Transactions on Speech and Audio Processing* (2000) (cf. p. 17).

[LLS05] T. Lemaire, S. Lacroix et J. Sola. "A practical 3D bearing-only SLAM algorithm". In : *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005 (cf. p. 85).

[LM97] F. Lu et E. Milios. "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans". In : *Journal of Intelligent and Robotic Systems* (1997) (cf. p. 84).

[LPD13] Z. Liu, Q. Pan et J. Dezert. "A new belief-based K-nearest neighbor classification method". In : *Pattern Recognition* (2013) (cf. p. 30, 38).

[LSB13] B. Lau, C. Sprunk et W. Burgard. "Efficient grid-based spatial representations for robot navigation in dynamic environments". In : *Robotics and Autonomous Systems* (2013) (cf. p. 105).

[LZJ02] L. Lu, H. Zhang et H. Jiang. "Content analysis for audio classification and segmentation". In : *IEEE Transactions on Speech and Audio Processing* (2002) (cf. p. 17, 18, 34).

[MCB11] J. Moras, V. Cherfaoui et P. Bonnifait. "Credibilist Occupancy Grids for Vehicle Perception in Dynamic Environments". In : *International Conference on Robotics and Automation*. 2011 (cf. p. 85, 130, 147).

[MGP09] S. Marat, N. Guyader et D. Pellerin. "Gaze prediction improvement by adding a face feature to a saliency model". In : *Recent advances in signal processing* (2009) (cf. p. 42).

[Mon+02] M. Montemerlo, S. Thrun, D. Koller et B. Wegbreit. "FastSLAM : A factored solution to the simultaneous localization and mapping problem". In : *AAAI/IAAI*. 2002 (cf. p. 84).

[MT07] N. Mitsou et C. Tzafestas. "Maximum Likelihood SLAM in Dynamic Environments". In : *IEEE International Conference on Tools with Artificial Intelligence(ICTAI)*. 2007 (cf. p. 83).

[Mur02] K. P. Murphy. "Hidden semi-markov models". In : *Unpublished notes* (2002) (cf. p. 143).

[Mur99] K. P. Murphy. "Bayesian Map Learning in Dynamic Environments." In : *NIPS*. 1999 (cf. p. 84).

[NC15] Q. Nguyen et J. Choi. "Selection of the Closest Sound Source for Robot Auditory Attention in Multi-source Scenarios". In : *Journal of Intelligent & Robotic Systems* (2015) (cf. p. 19).

[New+11] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges et A. Fitzgibbon. "KinectFusion : Real-time Dense Surface Mapping and Tracking". In : *IEEE International Symposium on Mixed and Augmented Reality*. 2011 (cf. p. 85).

[Orc14] A. Orcesi. *Détection et suivi d'objets à l'aide de données multimodales appliqués à la navigation robotique*. Rapp. tech. 2014 (cf. p. 69).

[Pap+09] G. Papandreou, A. Katsamanis, V. Pitsikalis et P. Maragos. "Adaptive Multimodal Fusion by Uncertainty Compensation With Application to Audio-visual Speech Recognition". In : *IEEE Transactions on Audio, Speech, and Language Processing* (2009) (cf. p. 46).

[Pic15] K. J. Piczak. "Environmental sound classification with convolutional neural networks". In : *IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. 2015 (cf. p. 19).

[PNDW96] D. Pagac, E. M. Nebot et H. Durrant-Whyte. "An evidential approach to probabilistic map-building". In : *Reasoning with Uncertainty in Robotics : International Workshop*. 1996 (cf. p. 127).

[Rab89] L. R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In : *Proceedings of the IEEE* (1989) (cf. p. 143).

[RB10] B. S. Reddy et O. A. Basir. "Concept-based evidential reasoning for multimodal fusion in human–computer interaction". In : *Applied Soft Computing* (2010) (cf. p. 51).

[RC13] T. Reineking et J. Clemens. "Evidential FastSLAM for grid mapping". In : *International Conference on Information Fusion (FUSION)*. 2013 (cf. p. 132).

[Rei11] T. Reineking. "Particle filtering in the Dempster–Shafer theory". In : *International Journal of Approximate Reasoning* (2011) (cf. p. 85, 132).

[Ren14] T. Reneking. "Belief functions : Theory and Algorithms". Thèse de doct. 2014 (cf. p. 132).

[RJ93] L. Rabiner et B. Juang. *Fundamentals of Speech Recognition*. 1993 (cf. p. 34).

[RMSL15] J. Rios-Martinez, A. Spalanzani et C. Laugier. "From Proxemics Theory to Socially-Aware Navigation : A Survey". In : *International Journal of Social Robotics* (2015) (cf. p. 42).

[RP97] V. Radova et J. Psutka. "An Approach to Speaker Identification Using Multiple Classifiers". In : *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 1997 (cf. p. 49).

[RRP07] E. Ramasso, M. Rombaut et D. Pellerin. In : *Symbolic and Quantitative Approaches to Reasoning with Uncertainty : 9th European Conference (ECSQARU)*. 2007 (cf. p. 143).

[Sau96] J. Saunders. "Real-time discrimination of broadcast speech/music". In : *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 1996 (cf. p. 17).

[SB15] J. Salamon et J. P. Bello. "Unsupervised feature learning for urban sound classification". In : *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015 (cf. p. 19, 21).

[SB16] J. Salamon et J. P. Bello. "Deep convolutional neural networks and data augmentation for environmental sound classification". In : *arXiv preprint arXiv :1608.04363* (2016) (cf. p. 19).

[SCI14] C. Siagian, C. K. Chang et L. Itti. "Autonomous mobile robot localization and navigation using a hierarchical map representation primarily guided by vision". In : *Journal of Field Robotics* (2014) (cf. p. 81, 87).

[Sha+14] L. Shao, J. Han, P. Kohli et Z. Zhang. *Computer vision and machine learning with RGB-D sensors*. Springer, 2014 (cf. p. 93).

[Sha76] G. Shafer. *A mathematical theory of evidence*. Princeton University Press, Princeton, 1976 (cf. p. 24).

[SI09] C. Siagian et L. Itti. "Biologically inspired mobile robot vision localization". In : *IEEE Transactions on Robotics* (2009) (cf. p. 81, 87).

[SJB14] J. Salamon, C. Jacoby et J. P. Bello. "A Dataset and Taxonomy for Urban Sound Research". In : *ACM International Conference on Multimedia*. 2014 (cf. p. 19, 21).

[SK94] P. Smets et Robert K. "The transferable belief model". In : *Artificial Intelligence* (1994) (cf. p. 24).

[Sme05] P. Smets. "Decision making in the TBM : the necessity of the pignistic transformation". In : *International Journal of Approximate Reasoning* (2005) (cf. p. 50).

[Spa15] A. Spalanzani. "Contribution à la navigation autonome en environnement dynamique et humain". Accreditation to supervise research. 2015 (cf. p. 7, 88).

[SPG12] G. Song, D. Pellerin et L. Granjon. "How different kinds of sound in videos can influence gaze". In : *International Workshop on Image Analysis for Multimedia Interactive Services*. 2012 (cf. p. 13).

[SPG13] G. Song, D. Pellerin et L. Granjon. "Different types of sounds influence gaze differently in videos". In : *Journal of Eye Movement Research* (2013) (cf. p. 13, 22, 42).

[SS01] B. Scholkopf et A. J. Smola. *Learning with Kernels : Support Vector Machines, Regularization, Optimization, and Beyond*. 2001 (cf. p. 16, 49).

[SWS05] C. G. M. Snoek, M. Worring et A. W. M. Smeulders. "Early Versus Late Fusion in Semantic Video Analysis". In : *ACM International Conference on Multimedia*. 2005 (cf. p. 43).

[TB96] S. Thrun et A. Bücken. "Integrating grid-based and topological maps for mobile robot navigation". In : *National Conference on Artificial Intelligence*. 1996 (cf. p. 81).

[TBF05] S. Thrun, W. Burgard et D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005 (cf. p. 7, 79, 81, 86).

[TC02] G. Tzanetakis et P. Cook. "Musical genre classification of audio signals". In : *IEEE Transactions on Speech and Audio Processing* (2002) (cf. p. 16–18, 34, 35).

[TH15] B. Tang et H. He. "ENN : Extended Nearest Neighbor Method for Pattern Recognition [Research Frontier]". In : *IEEE Computational Intelligence Magazine* (2015) (cf. p. 27).

[Thr+01] S. Thrun, D. Fox, W. Burgard et F. Dellaert. "Robust Monte Carlo localization for mobile robots". In : *Artificial intelligence* (2001) (cf. p. 86).

[Thr+04] S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto et E. Nebot. "Fastslam : An efficient solution to the simultaneous localization and mapping problem with unknown data association". In : *Journal of Machine Learning Research* (2004) (cf. p. 84).

[Thr98] S. Thrun. "Learning metric-topological maps for indoor mobile robot navigation". In : *Artificial Intelligence* (1998) (cf. p. 81).

[Tre+14] G. Trehard, Z. Alsayed, E. Pollard, B. Bradai et F. Nashashibi. "Credibilist Simultaneous Localization and Mapping with a LIDAR". In : *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014 (cf. p. 129, 130).

[Tre+15] G. Trehard, E. Pollard, B. Bradai et F. Nashashibi. "On line Mapping and Global Positioning for autonomous driving in urban environment based on Evidential SLAM". In : *Intelligent Vehicles Symposium-IV*. 2015 (cf. p. 129).

[VAA07] T. Vu, O. Aycard et N. Appenrodt. "Online Localization and Mapping with Moving Object Tracking in Dynamic Outdoor Environments". In : *IEEE Intelligent Vehicles Symposium*. 2007 (cf. p. 83).

[Vac+14] M. Vacher, B. Lecouteux, P. Chahuara, F. Portet, B. Meillon et N. Bonnefond. "The Sweet-Home speech and multimodal corpus for home automation interaction". In : *The 9th edition of the Language Resources and Evaluation Conference (LREC)*. 2014 (cf. p. 35, 172).

[VJ01] P. Viola et M. Jones. "Rapid object detection using a boosted cascade of simple features". In : *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2001 (cf. p. 53, 55, 56, 58).

[VJC16] D. Vaufreydaz, W. Johal et C. Combe. "Starting engagement detection towards a companion robot using multimodal features". In : *Robotics and Autonomous Systems* (2016) (cf. p. 7).

[VSA03] V. Vezhnevets, V. Sazonov et A. Andreeva. "A survey on pixel-based skin color detection techniques". In : *Proc. Graphicon*. 2003 (cf. p. 54).

[VSC15] E. Vincent, A. Sini et F. Charpillet. "Audio source localization by optimal control of a mobile robot". In : *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015 (cf. p. 19).

[Wal+08] M. Walters, S. Syrdal, K. Dautenhahn, R. Boekhorst et K. L. Koay. "Avoiding the uncanny valley : robot appearance, personality and consistency of behavior in an attention-seeking home scenario for a robot companion". In : *Autonomous Robots* (2008) (cf. p. 5).

[Wal+13] M. L. Walters, K. L. Koay, D. S. Syrdal, A. Campbell et K. Dautenhahn. "Companion robots for elderly people : Using theatre to investigate potential users' views". In : *IEEE RO-MAN*. 2013 (cf. p. 7).

[Whe+13] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard et J. McDonald. "Robust real-time visual odometry for dense RGB-D mapping". In : *IEEE International Conference on Robotics and Automation (ICRA)*. 2013 (cf. p. 85).

[XC06] H. Xu et T. Chua. "Fusion of AV Features and External Information Sources for Event Detection in Team Sports Video". In : *ACM Transactions on Multimedia Computing, Communications, and Applications* (2006) (cf. p. 46).

[Zad65] L. A. Zadeh. "Fuzzy sets". In : *Information and Control* (1965) (cf. p. 49).

[ZD98] L. M. Zouhal et T. Denoeux. "An evidence-theoretic k-NN rule with parameter optimization". In : *IEEE Transactions on Systems, Man, and Cybernetics* (1998) (cf. p. 30).

[ZLG09] Y. Zigel, D. Litvak et I. Gannot. "A Method for Automatic Fall Detection of Elderly People Using Floor Vibrations and Sound ;Proof of Concept on Human Mimicking Doll Falls". In : *IEEE Transactions on Biomedical Engineering* (2009) (cf. p. 19).

Résumé —

Dans ce travail, nous considérons le cas d'un robot mobile d'intérieur dont l'objectif est de détecter les humains présents dans l'environnement et de se positionner physiquement par rapport à eux, dans le but de mieux percevoir leur état. Pour cela, le robot dispose de différents capteurs (capteur RGB-Depth, microphones, télémètre laser). Des contributions de natures variées ont été effectuées :
Classification d'événements sonores en environnement intérieur : La méthode de classification proposée repose sur une taxonomie de petite taille et est destinée à différencier les marqueurs de la présence humaine. L'utilisation de fonctions de croyance permet de prendre en compte l'incertitude de la classification, et de labelliser un son comme « inconnu ».
Fusion audiovisuelle pour la détection de locuteurs successifs dans une conversation : Une méthode de détection de locuteurs est proposée dans le cas du robot immobile, placé comme témoin d'une interaction sociale. Elle repose sur une fusion audiovisuelle probabiliste. Cette méthode a été testée sur des vidéos acquises par le robot.
Navigation dédiée à la détection d'humains à l'aide d'une fusion multimodale : A partir d'informations provenant des capteurs hétérogènes, le robot cherche des humains de manière autonome dans un environnement connu. Les informations sont fusionnées au sein d'une grille de perception multimodale. Cette grille permet au robot de prendre une décision quant à son prochain déplacement, à l'aide d'un automate reposant sur des niveaux de priorité des informations perçues. Ce système a été implémenté et testé sur un robot Q.bo.
Modélisation crédibiliste de l'environnement pour la navigation : La construction de la grille de perception multimodale est améliorée à l'aide d'un mécanisme de fusion reposant sur la théorie des fonctions de croyance. Ceci permet au robot de maintenir une grille « évidentielle » dans le temps comprenant l'information perçue et son incertitude. Ce système a d'abord été évalué en simulation, puis sur le robot Q.bo.

Mots clés : Robots compagnons, fusion multimodale, navigation, fonctions de croyance

Abstract —

In this work, we consider the case of mobile robot that aims at detecting and positioning itself with respect to humans in its environment. In order to fulfill this mission, the robot is equipped with various sensors (RGB-Depth, microphones, laser telemeter). This thesis contains contributions of various natures :
Sound classification in indoor environments : A small taxonomy is proposed in a classification method destined to enable a robot to detect human presence. Uncertainty of classification is taken into account through the use of belief functions, allowing us to label a sound as "unknown".
Speaker tracking thanks to audiovisual data fusion : The robot is witness to a social interaction and tracks the successive speakers with probabilistic audiovisual data fusion. The proposed method was tested on videos extracted from the robot's sensors.
Navigation dedicated to human detection thanks to a multimodal fusion : The robot autonomously navigates in a known environment to detect humans thanks to heterogeneous sensors. The data is fused to create a multimodal perception grid. This grid enables the robot to chose its destinations, depending on the priority of perceived information. This system was implemented and tested on a Q.bo robot.
Credibilist modelization of the environment for navigation : The creation of the multimodal perception grid is improved by the use of credibilist fusion. This enables the robot to maintain an evidential grid in time, containing the perceived information and its uncertainty. This system was implemented in simulation first, and then on a Q.bo robot.

Keywords : Companion robots, multimodal fusion, navigation, belief functions