UNIVERSITÉ DU QUÉBEC À MONTRÉAL

CONCEPTION ET INTÉGRATION D'ALGORITHMES DE RECONNAISSANCE DE PLAN À DESTINATION D'UN ROBOT MOBILE ASSISTANT

THÈSE
PRÉSENTÉE
COMME EXIGENCE PARTIELLE
DU DOCTORAT EN INFORMATIQUE

PAR
JEAN MASSARDI

AOÛT 2021

UNIVERSITÉ DU QUÉBEC À MONTRÉAL
Service des bibliothèques

Avertissement

La diffusion de cette thèse se fait dans le respect des droits de son auteur, qui a signé le formulaire *Autorisation de reproduire et de diffuser un travail de recherche de cycles supérieurs* (SDU-522 – Rév.04-2020). Cette autorisation stipule que «conformément à l'article 11 du Règlement no 8 des études de cycles supérieurs, [l'auteur] concède à l'Université du Québec à Montréal une licence non exclusive d'utilisation et de publication de la totalité ou d'une partie importante de [son] travail de recherche pour des fins pédagogiques et non commerciales. Plus précisément, [l'auteur] autorise l'Université du Québec à Montréal à reproduire, diffuser, prêter, distribuer ou vendre des copies de [son] travail de recherche à des fins non commerciales sur quelque support que ce soit, y compris l'Internet. Cette licence et cette autorisation n'entraînent pas une renonciation de [la] part [de l'auteur] à [ses] droits moraux ni à [ses] droits de propriété intellectuelle. Sauf entente contraire, [l'auteur] conserve la liberté de diffuser et de commercialiser ou non ce travail dont [il] possède un exemplaire.»

REMERCIEMENTS

Cette thèse n'aurait pas été possible sans le support et l'appui de bon nombre de personnes. Le premier est bien sûr Éric Beaudry, qui m'a supervisé au cours de ces cinq longues années. Au rang de ceux à remercier, il y a bien entendu mes collègues et camarades de laboratoire du GDAC du 4e étage, Johan, Jonathan, Jaël, Éric et Gabrielle, dont la bonne humeur et l'humour, souvent bien au-delà des frontières du bon goût, ont rendu ce périple beaucoup plus supportable. J'en profite aussi pour remercier tous les stagiaires accueillis au cours des années qui ont apporté leurs compétences et leur enthousiasme à mes travaux. J'aimerais aussi remercier les autres thésards ayant partagé ces années avec moi, Ange, Michael et surtout Sasha. Sasha sans toi, ta fougue, ton dynamisme et ton manque absolu de limites, je ne serais pas allé bien loin.

Mes remerciements vont aussi à ma famille et d'abord à mes frères et soeurs, Paul, Louise et Marie. À mes parents ensuite, Thierry et Isabelle, sans quoi rien de tout ça ne serait possible. Ils m'ont supporté, à tous les sens du terme, tout au long de l'aventure, dans le pire et le meilleur. Je suis immensément fier d'être votre fils.

Je tiens à remercier mes amis et en premier les anciens de LDS, surtout Daniel et Sylvain. Ensuite Mathieu, je ne t'ai pas oublié, sans toi tout ça ne serait rien. Enfin Olivier, tu es l'ami le plus intime, le plus fidèle et le plus pénible que je n'aurais jamais espéré avoir. Tu as été un roc sur lequel j'ai pu me reposer.

Cette thèse a été rendue possible grâce au soutien financier d'*Aging Gracefully across Environments using Technology to Support Wellness, Engagement and Long Life NCE Inc.* (AGE-WELL).

DÉDICACE

À Amandine.

TABLE DES MATIÈRES

LISTE DES TABLEAUX vii
LISTE DES FIGURES ix
RÉSUMÉ xi
INTRODUCTION 1
CHAPITRE I RECONNAISSANCE DE PLAN HIÉRARCHIQUE PARTIELLEMENT ORDONNÉ 10
1.1 État de l'art en plans hiérarchiques partiellement ordonnés 13
1.1.1 Définition des plans hiérarchiques partiellement ordonnés . . . 14
1.1.2 Modèle de décision 16
1.1.3 Algorithmes de résolution 17
1.2 Observabilité imparfaite 25
1.3 Approches par filtre à particules 28
1.4 Application aux plans hiérarchiques 30
1.4.1 Initialisation 34
1.4.2 Filtrage et ré-échantillonnage 35
1.5 Évaluation 38
1.5.1 Domaine 39
1.5.2 Précision 39
1.5.3 Rapidité 41
1.5.4 Robustesse aux bruits 42
1.6 Discussion 45
1.7 Conclusion 47
CHAPITRE II RECONNAISSANCE DES ACTIVITÉS QUOTIDIENNES 49
2.1 État de l'art en reconnaissance des ADLs 51

2.1.1 Affordance d'objet 51
2.1.2 Suivi des mouvements 54
2.2 Architecture de PARC 56
2.2.1 Reconnaissance d'objets et de mains 58
2.2.2 Détection des affordances 59
2.3 Évaluation 62
2.3.1 Domaines 63
2.3.2 Intégration entre reconnaissance de plan et reconnaissance d'activité 64
2.3.3 Reconnaissance d'ADLs 66
2.4 Discussion 68
2.5 Conclusion 70
CHAPITRE III DÉTECTION D'ANOMALIES DANS L'EXÉCUTION DE PLANS 73
3.1 État de l'art en détection d'anomalies 75
3.2 Approche par reconnaissance d'objectifs 78
3.3 Approche par prédiction d'observations 84
3.4 Évaluation 85
3.4.1 Domaines 85
3.4.2 Résultats 87
3.5 Discussion 93
3.6 Conclusion 96
CHAPITRE IV RECONNAISSANCE D'OBJECTIF ACTIVE 99
4.1 État de l'art en planification et reconnaissance d'objectif 102
4.2 Présentation des modèles MDPs 103
4.2.1 MDPs 104
4.2.2 POMDPs 106
4.3 IAMP framework 108

4.4 Extension à la reconnaissance de plan 109
4.4.1 Définitions 110
4.4.2 MOMDP Framework pour l'AGR 112
4.4.3 Exemple 114
4.5 Évaluation 117
4.5.1 Problème 1 : Détection de trajets avec limitation de ressources 117
4.5.2 Problème 2 : positionnement d'un robot mobile 119
4.6 Discussion 122
4.7 Conclusion 124
CONCLUSION 125
RÉFÉRENCES 129

LISTE DES TABLEAUX

Tableau Page

1.1 Temps d'exécution moyen en secondes pour PHATT, SLIM et un filtre à particules de 500 particules (PF-500) 41

1.2 Précision moyenne finale en pourcentage pour un filtre à particules avec 500 particules. 44

2.1 Utilisation du processeur et de la mémoire 67

3.1 Détection d'anomalies en utilisant l'approche par prédiction d'objectifs pour un domaine avec 10 actions - cas avec des anomalies dans la séquence 88

3.2 Détection d'anomalies en utilisant l'approche par prédiction d'objectif pour un domaine avec 100 actions - cas avec anomalies dans la séquence 89

3.3 Détection d'anomalies en utilisant l'approche par prédiction d'objectifs pour un domaine avec 10 actions - cas sans anomalies dans la séquence 90

3.4 Détection d'anomalies en utilisant l'approche par prédiction d'objectif pour un domaine avec 100 actions - cas sans anomalies dans la séquence 91

3.5 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 10 actions - cas avec anomalies dans la séquence 92

3.6 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 100 actions - cas avec anomalies dans la séquence 93

3.7 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 10 actions - cas sans anomalies dans la séquence 94

3.8 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 100 actions - cas sans anomalies dans la séquence 95

3.9 Détection d'anomalies en utilisant une approche floue - cas avec anomalies dans la séquence 96

3.10 Détection d'anomalies en utilisant une approche floue - cas sans anomalies dans la séquence 97

4.1 Temps de calcul moyen, nombre d'observations moyennes, vitesse de convergence moyenne en mode actif et vitesse de convergence en cas d'observation complète pour la détection de trajets 118

4.2 Temps de calcul, états de croyance et espérance de récompense pour le problème du robot mobile 121

LISTE DES FIGURES

Figure		Page
0.1	Processus de reconnaissance de plan	2
0.2	Un robot BEAM +. Source : (ROBOTS IEEE 2015)	5
1.1	Exemple simple d'une bibliothèque de plans	15
1.2	Exemple d'application du filtre à particules	37
1.3	Précision de la reconnaissance de plans en fonction de la complétion du plan pour 50, 100, 250 et 500 particules	40
1.4	Précision de la reconnaissance de plan en fonction de la complétion du plan pour 250, 500 et 1000 particules pour un bruit de 0%, 10%, 20% et 30%	43
1.5	Exemple de repères pour la réalisation d'un objectif. Source : (Pereira *et al.*, 2020a)	46
2.1	Représentation des liens d'affordance entre les objets, les actions et leurs effets. Source : (Lopes *et al.*, 2007)	52
2.2	Utilisation de puces RFID dans l'environnement et sur les personnes observés. Source : (Fortin-Simard *et al.*, 2015)	53
2.3	Exemple de renaissance d'ADL par analyse de mouvements et de poses. Source : (Xia *et al.*, 2015)	54
2.4	Architecture de PARC	57
2.5	Exemple de partitionnement des mesures de profondeurs. Les zones en rouge sont retirées, car jugées aberrantes. Les zones en bleu sont considérées comme ayant une variation acceptable.	59
2.6	Turtlebot avec une Intel RealSense D-435 comme prototype pour PARC	62
2.7	Exemple d'une prise de vue de PARC dans les conditions d'expérimentation avec les balles de couleurs	63

2.8 Précision en fonction de la complétion du plan 65

2.9 Matrice de confusion de la détection d'activité. Missing indique les activités de bas niveau non-reconnu par le système. 68

2.10 Matrice de confusion de la reconnaissance de plan 69

4.1 Vision subjective du robot mobile avec caméra embarquée 100

4.2 Exemple d'un MDP simple 106

4.3 Représentation du processus de reconnaissance d'objectif active par MOMPD 114

4.4 Exemple de problème d'AGR : reconnaissance d'objectifs en limitant le nombre d'observations 115

4.5 Représentation du problème d'AGR pour un robot mobile 120

RÉSUMÉ

Cette thèse présente des contributions dans le domaine de la reconnaissance de plan, un sous-domaine de l'Intelligence Artificielle. Il s'agit pour un agent observateur de reconnaître les objectifs que cherche à réaliser un agent observé. La finalité des travaux présentés ici est d'être appliqués sur des robots assistants mobiles d'aide pour les personnes âgées.

Les travaux présentés ici s'intéressent aux cas de la reconnaissance de plan dans le cas d'observation bruitée, c'est-à-dire où les observations ne correspondent pas nécessairement à ce qu'est la réalité, et dans un contexte temps réel. Une solution au problème introduit dans cette thèse est l'utilisation de filtre à particules sur des bibliothèques de plans, dans lesquelles une modélisation du bruit est représentée. Cette solution est intégrée à PARC, un système hybride de reconnaissance de plan et d'activité, conçu pour être temps réel et embarqué sur un robot. Dans PARC, les actions de bases sont décrites comme des interactions entre un agent observé et une main, le résultat est ensuite envoyé comme observation au système de reconnaissance de plan utilisant des filtres à particules. Deux extensions de ces travaux sont proposées pour être adaptées à un autre problème, la détection d'anomalie. La première extension utilise les prédictions des objectifs pour reconnaître les actions anormales, la seconde extension utilise les prédictions des observations futures elle aussi pour reconnaître les actions anormales. Enfin des contributions sur un problème de planification d'observations dans le cas de la reconnaissance de plan, appelé reconnaissance d'objectif active, sont présentées.

Mots-clés : Intelligence artificielle ; reconnaissance de plan ; planification automatique ; temps réel

INTRODUCTION

L'Intelligence Artificielle (IA) est un domaine de recherche visant à doter les machines de capacité de résolution de problèmes de manière autonome. Un des sous-domaines de l'IA est la planification automatique. Ce sous-domaine étudie les problèmes de génération de plan, c'est-à-dire générer un ensemble d'actions, par exemple une séquence d'actions ordonnées, visant à atteindre un objectif prédéfini. La reconnaissance de plan, un autre sous-domaine de l'IA, consiste à résoudre le problème inverse de la planification (Kautz, 1987). Dans sa forme la plus simple, la reconnaissance de plan consiste à reconnaître l'objectif d'un agent à partir d'une séquence d'actions observées. La figure 0.1, présente le processus de reconnaissance de plan. L'agent observé a un but. Ce but se traduit par un plan qui l'amène à réaliser une séquence d'actions. Ces actions produisent des observations qui sont analysées par l'observateur. Dans l'exemple de la figure 0.1, le but de l'agent est de se faire du thé, ce qui se traduit par une séquence d'actions commençant ici par attraper une tasse.

Le domaine de la reconnaissance de plan peut se diviser en deux grands axes de recherche : la reconnaissance d'activité et la reconnaissance de but et d'intention. Dans la littérature, le terme de «reconnaissance de plan» peut aussi bien désigner la reconnaissance d'activité que la reconnaissance de but ou d'intention.

La reconnaissance d'activité se concentre sur la découverte, la modélisation et l'identification d'actions simples ou complexes exécutées par l'agent observé. Il existe plusieurs façons de détecter des activités. La plus simple est l'usage de capteurs installés dans l'environnement pour détecter des actions, par exemple un

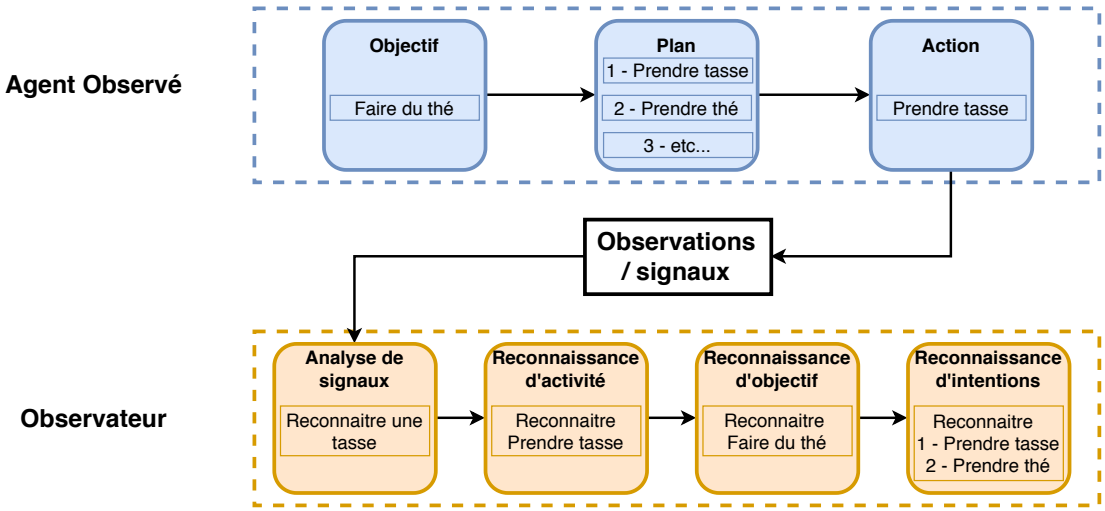


Figure 0.1 Processus de reconnaissance de plan

capteur de pression sous un objet pour détecter sa prise. En robotique, la reconnaissance d'activité s'appuie principalement sur des techniques d'analyse et de reconnaissance d'images et de signaux (Sukthankar *et al.*, 2014). Dans l'exemple de la figure 0.1, l'observateur reconnaît une tasse, ce que la reconnaissance d'activité va traduire par l'action la plus probable de l'agent observé, c'est-à-dire prendre une tasse.

La reconnaissance d'objectif et d'intention quant à elle se concentre sur l'analyse de tâches et d'activités de plus haut niveau en utilisant des techniques de raisonnement. La distinction entre reconnaissance d'activité et reconnaissance de but/intention se fait en fonction de la complexité des activités observées et de la nature de l'algorithme utilisé.

La reconnaissance de plan dans son ensemble est une tâche complexe dont la figure 0.1 illustre un cas général théorique. Dans la pratique, chacune des étapes décrites peut contenir des anomalies. Du côté de l'agent observé, on peut trouver ces anomalies dans le plan, l'agent peut choisir d'exécuter un plan qui ne permet

pas d'atteindre ses objectifs, ou dans les actions, l'agent peut mal exécuter une action particulière. En ce qui concerne l'observateur, chacune des étapes de reconnaissance peut se faire dans un environnement bruité et par conséquent pouvant contenir des erreurs.

La reconnaissance de plan peut être séparée en deux catégories (Sukthankar *et al.*, 2014) : la reconnaissance d'objectif (*goal recognition*) et la reconnaissance d'intentions (*intent recognition*). La reconnaissance de but vise à reconnaître la finalité des actions entreprises par un agent alors que la reconnaissance d'intention cherche à estimer la suite d'actions probables que va entreprendre un agent dans le futur. Habituellement, la reconnaissance d'intention est vue comme étant une extension de la reconnaissance de but. Une approche typique pour la reconnaissance d'intention est de commencer par une étape de reconnaissance de but, puis une fois les buts identifiés, utiliser la planification afin de prévoir les actions futures possibles de l'agent observé. Toujours dans la figure 0.1, le processus de reconnaissance de but pourrait inférer, à partir de l'action prendre une tasse, le but de l'agent observé. À partir de ce but, l'observateur peut retrouver une séquence d'actions probables que l'agent observé réalisera pour atteindre ce but. Dans ce document, le terme de reconnaissance de plan sera utilisé pour désigner le domaine recouvrant l'étude de la reconnaissance de but et de la reconnaissance d'intention.

Enfin, la reconnaissance de plan peut se faire dans trois contextes (Sukthankar *et al.*, 2014) :

1. De manière externe (*keyhole*), quand il n'y pas d'interactions possibles entre l'agent observé et l'observateur. Il s'agit du contexte d'application de la reconnaissance de plan le plus utilisé, car il s'agit en général de la forme la plus simple.
2. De manière coopérative (*intended*), quand l'agent observé et l'observateur

coopèrent. Par exemple, l'agent observé est conscient qu'il est observé et il peut volontairement exécuter des actions ayant pour but de s'assurer que ses intentions soient bien comprises par l'observateur. On retrouve ce contexte dans le cas de certains robots assistants.

3. De manière hostile, aussi appelée reconnaissance des adversaires (*adversarial*), quand les agents sont en opposition. Par exemple, l'agent observé peut cacher ou exécuter des actions pour tromper l'observateur. Ce contexte est surtout utilisé dans le cadre de la reconnaissance de plan en stratégie, souvent combinée avec des éléments de théorie des jeux. On retrouve ce contexte dans des applications militaires ou en sécurité informatique.

Pour faire face à la problématique du vieillissement de la population des pays développés, notamment la population canadienne, de nombreux organismes proposent d'utiliser les avancées technologiques récentes, dont en Intelligence Artificielle, pour améliorer les conditions de vie des personnes âgées. Parmi ces organismes, on retrouve le réseau d'excellence AGE-WELL NCE (*Aging Gracefully across Environments using Technology to Support Wellness, Engagement and Long Life NCE Inc.*), un organisme pan-canadien. AGE-WELL NCE finance plusieurs projets dont le *Workpackage* 3.1, aussi appelé projet VIGIL, un projet visant la conceptions de technologies pouvant être embarqué sur des robots assistants (Laniel *et al.*, 2017b). Ce projet est une collaboration de plusieurs chercheurs établis dans quatre universités : l'université de Toronto, l'université de Sherbrooke, l'université Western Ontario et l'université du Québec a Montréal.

Le projet VIGIL vise à concevoir des plateformes mobiles à faible coût destinées à l'assistance des personnes âgées à leur domicile. L'un des principes de VIGIL est de maintenir les capteurs sur la plateforme et ainsi éviter de les déployer dans l'environnement. Ce choix a été fait pour deux raisons : (1) faciliter l'acceptabilité de ces technologies en ne modifiant pas le lieu de vie des personnes âgées, (2)

minimiser le coût en évitant la multiplication des capteurs comme on peut le trouver en domotique (Laniel *et al.*, 2017a).

L'un des robots prototypes réalisés dans le cadre du projet VIGIL est SAM. Il s'agit d'un robot Beam + sur lequel des capteurs et de la puissance de calcul supplémentaire ont été ajoutés. Il est destiné à servir à la foi comme robot de téléprésence et comme assistant cognitif pour les activités quotidiennes.

Figure 0.2 Un robot BEAM +. Source : (ROBOTS IEEE 2015)

Les activités quotidiennes, aussi appelées ADL (*Activities for Daily Living*), sont l'ensemble des activités qu'une personne peut faire dans sa vie de tous les jours, notamment à domicile, par exemple se faire une boisson chaude, faire le ménage ou encore se préparer pour sortir. Afin de pouvoir apporter une aide cognitive, un système assistant doit dans un premier temps être capable d'identifier ces ADLs

et dans un second temps identifier les cas où une aide est nécessaire, c'est-à-dire où des comportements sont anormaux (Giroux et Pigot, 2012). On peut avoir par exemple le cas d'une personne qui se prépare une collation en cuisine mais qui oublie d'éteindre le feu. Le système peut apporter une aide en rappelant de réaliser cette dernière action, c'est-à-dire éteindre le feu.

La reconnaissance des ADLs peut se faire en utilisant deux stratégies : (1) une approche utilisant la reconnaissance d'activité (Aggarwal et Ryoo, 2011), (2) une approche utilisant la reconnaissance de plan combinée avec de la reconnaissance d'activité (Granada *et al.*, 2017). La reconnaissance d'activité utilisée seule a plusieurs limitations. D'abord, le fait de décrire chaque action de manière isolée rend l'approche peu expressive, notamment dans le cas des activités complexes et composées. Comme chaque action est isolée, cette approche ne permet pas de décrire des successions d'actions, or ce sont les successions qui permettent de reconnaître des erreurs d'exécution, de raisonnement ou bien des oublis. Ce sont ces erreurs qui donnent des informations sur de possibles problèmes cognitifs (Giroux et Pigot, 2012).

L'objectif de cette thèse est de proposer des solutions à la problématique de la reconnaissance de plan dans le cas d'un robot mobile dont les capteurs se trouvent intégrés sur le robot et dans le cas de la reconnaissance des ADL pour des personnes pouvant être atteintes de problèmes cognitifs.

Cette problématique induit plusieurs sous-problèmes. Tous d'abord la présence d'une caméra unique sur un robot rend la reconnaissance d'activité moins précise et donc plus sujette à fournir des informations incomplètes ou erronées (Wu *et al.*, 2007) (Koppula *et al.*, 2013). Ensuite, la reconnaissance de plan doit pouvoir se faire en temps réel sur des activités potentiellement complexes. Cet aspect est rendu plus difficile par la contrainte posée par le bruit, c'est-à-dire les informa-

tions incomplètes ou erronées, dans la reconnaissance d'activité (Kabanza *et al.*, 2013) (Vattam et Aha, 2015a). Un autre sous-problème vient de la détection de comportements anormaux. Là où il existe quelques approches, notamment en domotique, il y a un manque d'approches plus globales (Giroux et Pigot, 2012). Enfin, comme le robot est mobile, il se pose le problème de son positionnement dans l'espace afin d'obtenir les meilleures observations possibles (Kabanza *et al.*, 2010).

Cette thèse propose les contributions suivantes, concentrées sur le domaine de la reconnaissance de plans et ses applications :

1. Une définition formelle du bruit dans les observations pour la reconnaissance de plan hiérarchique utilisant une formalisation issue des grammaires non contextuelles.
2. Un algorithme de résolution temps réel robuste au bruit pour ces mêmes plans hiérarchiques. Ces deux premières contributions ont été présentées à l'*International Conference on Automated Planning and Scheduling* (ICAPS) 2019 (Massardi *et al.*, 2019). Cette présentation fut aussi accompagnée d'une démonstration technique durant la conférence.
3. Une intégration de la reconnaissance de plan et de la reconnaissance d'activité sur flux vidéo en une seule application. Ce travail a été présenté à l'International Conference on Robotics and Automation (ICRA) 2020 (Massardi *et al.*, 2020) et a fait l'objet d'une démonstration technique au workshop Plan, Activity, and Intent Recognition (AAAI-PAIR) 2020.
4. Un modèle de reconnaissance de comportements anormaux en utilisant toujours le formalisme de plans hiérarchiques. Ce travail est en cours de publication et a été présenté au workshop *Robots for Health and Elderly Care* de l'*International Conference on Intelligent Robots and Systems* IROS-RoboHEC 2020 (Massardi et Beaudry, 2020).

5. Une approche de reconnaissance de plan active, c'est à dire avec un observateur actif visant à améliorer ses prédictions. Ce travail a été accepté pour être présenté au workshop *Plan, Activity, and Intent Recognition* de la conférence de l'*Association for the Advancement of Artificial Intelligence*(AAAI-PAIR) 2021.

Cette thèse suit la structure d'une thèse par insertion d'articles sans pour autant l'être. Chacun des chapitres est basé sur des travaux présentés à des conférences à comité de relecture. Ces chapitres sont des extensions de la traduction des articles présentés.

Dans le Chapitre 1 sont présentés les travaux concernant la reconnaissance de plan hiérarchiques. Ce chapitre introduit une définition mathématique des plans hiérarchiques. Il s'en suit une discussion des principales techniques pour résoudre le problème de la reconnaissance de plan utilisant ce formalisme. Nous présentons ensuite une extension du modèle permettant l'expression du bruit ainsi que la technique basée sur les filtres à particule que nous avons développé pour adresser ce problème.

Dans le Chapitre 2 est présentée l'intégration de la reconnaissance de plan avec la reconnaissance d'activité afin de faire un système de reconnaissance des ADL. Ce chapitre commence par un état de l'art des techniques utilisées en reconnaissance d'activités sur flux vidéos avant de présenter l'architecture de la solution retenue qui exploite les interactions avec les objets faits par un humain.

Le Chapitre 3 se concentre sur une application de notre approche de reconnaissance de plan sur le problème de reconnaissance de comportements anormaux. Après une présentation des utilisations combinées de la reconnaissance de plan et de la reconnaissance de comportements anormaux, nous donnons une définition formelle de ces comportements anormaux en utilisant le formalisme présenté dans

le Chapitre 1. Nous présenterons ensuite des approches issues de l'adaptation de notre filtre à particules à ce type de problèmes.

Le Chapitre 4 présente nos travaux concernant la reconnaissance de plan active, c'est-à-dire la planification des observations par l'observateur afin d'améliorer sa reconnaissance du plan de l'agent observé. Ce chapitre commence par une introduction au processus de décisions Markoviens avant de s'intéresser à une méthode de planification permettant de prendre en compte les intentions d'agents extérieurs. Nous présenterons dans ce chapitre une adaptation de cette méthode permettant de supporter la reconnaissance de plan.

Une conclusion s'en suit.

CHAPITRE I

RECONNAISSANCE DE PLAN HIÉRARCHIQUE PARTIELLEMENT ORDONNÉ

Les premiers travaux sur la reconnaissance de plan, notamment ceux de Kautz et Allen proposent de s'adresser à cette problématique par le biais des inférences logiques (Kautz, 1987) . Kautz et Allen, dans leurs travaux, utilisent des graphes de plans non ordonnés. Ceux-ci sont présentés sous forme d'arbres, chacun ayant comme racine un objectif. Dans cette représentation, une hypothèse de plan valide est un graphe de plan recouvrant toutes les observations. Charniak présente le problème de reconnaissance de plan comme un problème d'abduction formelle (Charniak et Goldman, 1993). L'abduction formelle telle que définie par Peirce (Peirce, 1903) est la tâche qui consiste à inverser une implication afin de retrouver une cause liée à une conséquence, par exemple inférer la probabilité d'un feu en observant de la fumée. Il s'agit d'une forme de logique imparfaite. Pour Charniak, comme il s'agit d'abduction formelle, la meilleure approche consiste à utiliser un raisonnement bayésien, particulièrement adapté à ce genre de problèmes. La plupart des travaux récents utilisent une approche combinant à la fois une confrontation d'hypothèses de plans à des d'observations, et un raisonnement bayésien sur ces hypothèses (Sukthankar *et al.*, 2014).

La confrontation des hypothèses à des observations peut se faire en utilisant deux

approches : (1) l'utilisation de bibliothèques de plans (Geib et Goldman, 2009) et (2) l'utilisation de la planification (*Plan recognition as planning*) (Ramırez et Geffner, 2009). Dans la première approche, la reconnaissance de plan se fait en utilisant des collections d'actions, appelées bibliothèques de plans. Ces collections d'actions peuvent être plus ou moins explicites et plus ou moins ordonnées en fonction de la représentation choisie préalablement. Dans la seconde approche, un planificateur automatique est utilisé pour générer un ensemble de plans pour décrire le comportement d'un agent, en fonction des objectifs connus et des observations en cours.

Les approches par bibliothèques de plans souffrent de trois limitations importantes :

1. elles nécessitent la construction à priori d'une bibliothèque de plans (Sukthankar *et al.*, 2014) (Geib et Kantharaju, 2018). Celles-ci sont généralement non-exhaustive et peuvent être sujette à des erreurs humaines de modélisation ;
2. Il y a un compromis à faire entre expressivité et utilisation de ressources (Wu *et al.*, 2007) (Geib et Goldman, 2009) ;
3. peu de travail a été réalisé afin de gérer les observations imparfaites à l'exception des observations manquantes (Vattam et Aha, 2015b) (Geib et Kantharaju, 2018).

En ce qui concerne les approches utilisant des planificateurs, elles souffrent elles aussi de trois défauts :

1. elles nécessitent la création d'un domaine de planification (Ramırez et Geffner, 2009). Cela consiste dans le cas de planification classique à déterminer les actions avec leurs préconditions, c'est-à-dire l'état du monde permettant leur utilisation, et leurs effets ;
2. elles supposent que l'agent observé se comporte de manière rationnelle et

optimale. On sous-entend ici que l'agent observé va chercher à réaliser son ou ses objectifs en utilisant un plan, c'est-à-dire une suite d'action, minimisant une fonction de coût préalablement définie ;

3. l'utilisation de planificateurs de manière répétée entraîne des problèmes d'extensibilité.

Au vu de notre application, c'est-à-dire la reconnaissance de plan pour des activités quotidiennes pour des personnes âgées en situation possible de détresse cognitive, nous proposons une stratégie utilisant des bibliothèques de plans plutôt que la planification. La raison principale est que l'hypothèse de comportement rationnel et optimal ne sera vraisemblablement pas respectée. Ici on peut supposer qu'une personne à domicile va plutôt agir par habitude, en utilisant des plans routiniers plutôt que des plans optimaux.

Notre système reprend le formalisme de bibliothèques de plans proposé par Geib et Goldman (Geib et Goldman, 2009). Celui-ci consiste à décrire les plans sous forme d'arbres d'actions hiérarchiques partiellement ordonnées, aussi appelé plans hiérarchiques partiellement ordonnés. Ces arbres sont représentés sous forme de grammaires non contextuelles. Ce formalisme a plusieurs caractéristiques qui le rendent pertinent pour notre étude. Tout d'abord il est à la fois compact et expressif. Ensuite de nombreuses approches de la reconnaissance de plan ont été proposées en utilisant ce formalisme (Geib et Goldman, 2001) (Geib *et al.*, 2008) (Kabanza *et al.*, 2013) (Bisson *et al.*, 2015) (Mirsky et Gal, 2016). Il faut cependant noter qu'à notre connaissance, ce formalisme n'a pas été utilisé dans le cas d'applications réelles, contrairement à des modèles plus simples et moins expressifs tels que les réseaux bayésiens dynamiques (Wu *et al.*, 2007) ou encore SBR (*Symbolic Behavior Recognition*) (Granada *et al.*, 2017). Deux raisons peuvent expliquer l'absence d'applications : (1) la plupart des approches ne sont pas temps réel, (2) les approches liées à ce formalisme supportent mal les observations im-

parfaites.

Dans ce chapitre, nous présentons nos travaux concernant la reconnaissance de plan utilisant des bibliothèques de plans sous forme d'arbres d'actions hiérarchiques partiellement ordonnées. Pour faire face aux deux problématiques que sont la gestion des observations imparfaites et l'utilisation en temps réel, nous proposons d'utiliser des filtres à particules (*Particule filter*), une méthode d'exploration d'hypothèse par échantillonnage.

Ce chapitre est principalement basé sur notre article présenté à l'International Conference on Automated Planning and Scheduling (ICAPS) (Massardi *et al.*, 2019).

Ce chapitre se compose des sections suivantes. Dans la section 1.1 nous présentons le formalisme utilisé, notamment ses définitions et quelques approches pertinentes de reconnaissance de plans en découlant. Dans la section, 1.2 nous présentons notre modèle permettant de représenter les observations imparfaites, celui-ci étant la première contribution originale de notre recherche. Dans les sections 1.3 et 1.4, nous présentons les filtres à particules et leurs applications utilisant le formalisme de Geib et Goldman enrichit de notre définition des observations imparfaites. À la section 1.5, nous présentons une évaluation de notre approche sur les jeux de tests standards utilisant ce formalisme. Enfin dans la section 1.6, nous présentons une discussion sur nos travaux, notamment sur la problématique de construire des bibliothèques de plans.

1.1 État de l'art en plans hiérarchiques partiellement ordonnés

Les bibliothèques de plans hiérarchiques utilisés par Geib et Goldman sont des règles de production auxquelles on adjoint des contraintes d'ordonnancements partiels. Cette représentation est très inspirée du fonctionnement des HTNs (*Hierar-*

chical Task Network), des formalismes de planification (Erol, 1996). La différence principale entre les HTNs et les bibliothèques de plans hiérarchiques vient du fait que dans les bibliothèques de plans hiérarchiques il n'y a pas de préconditions et d'effets sur les actions. Formellement parlant, il s'agit d'une extension des grammaires non contextuelles (*CFG pour Context Free Grammar*) (Chomsky, 1956).

1.1.1 Définition des plans hiérarchiques partiellement ordonnés

En utilisant la définition de (Geib et Goldman, 2009) :

Définition 1.1. *Une Bibliothèque de plans PL est un tuple* $PL = (A, NT, G, P)$ *ou :*

- A *est un ensemble fini de symboles terminaux ;*
- NT *est un ensemble fini de symboles non terminaux ;*
- $G \subset NT$ *est un ensemble fini d'objectifs ;*
- P *est un ensemble fini de règles de production de la forme* $\alpha \rightarrow S, \sigma$*, avec* $\alpha \in NT$*,* S *un ensemble de symboles de* $A \cup NT$ *et* σ *un ordonnancement partiel de* S*.*

L'ensemble des symboles terminaux A correspond à l'ensemble des actions observables que peut réaliser un agent observé. Par la suite, nous utiliserons ce terme d'action plutôt que le terme symboles terminaux pour faciliter la lecture.

Un ordonnancement partiel σ est une contrainte de la forme (i, j),où la i^e action S_i doit être exécutée avant la j^e action S_j pour pouvoir accomplir la tâche α.

En utilisant ce formalisme, les plans peuvent être décrits comme des arbres où l'objectif principal se trouve à la racine et les actions de bas niveau, c'est-à-dire généralement directement observable, se trouvent au niveau des feuilles. Les noeuds intermédiaires représentent les sous-objectifs.

Les règles de productions permettent de décomposer un symbole non terminal en une séquence partiellement ordonnée de symboles pouvant être non terminaux ou terminaux.

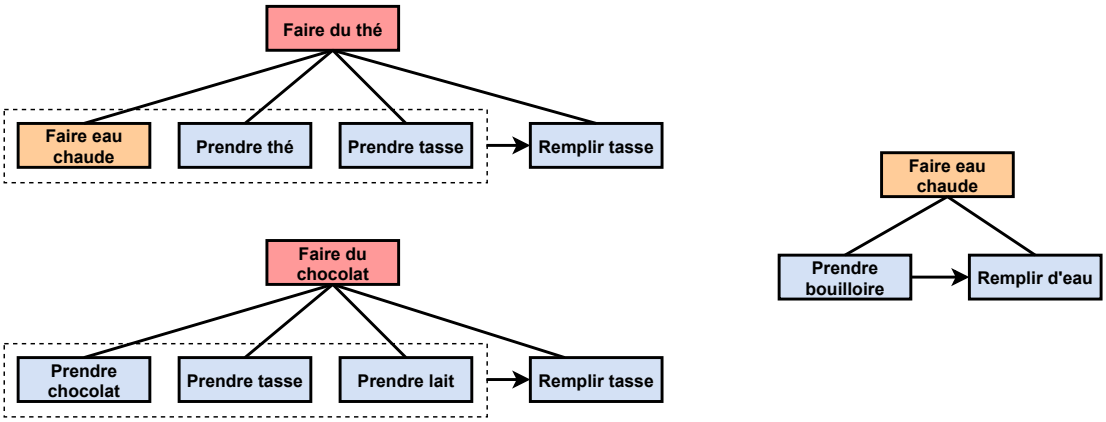


Figure 1.1 Exemple simple d'une bibliothèque de plans

La figure 1.1 représente une bibliothèque de plans simples avec deux objectifs en racine. Ces deux objectifs décrivent des séquences d'actions permettant de réaliser une boisson, du thé pour l'un et du chocolat pour l'autre. Dans cet exemple, les actions *Prendre thé*, *Prendre tasse* et le sous-objectif *Faire eau chaude* peuvent être réalisés dans n'importe quel ordre, mais ils doivent être réalisés avant l'action *Remplir tasse* pour réaliser l'objectif *Faire du thé*. L'objectif *Faire du chocolat* et le sous-objectif *Faire eau chaude* peuvent être décomposés avec la même logique.

En projetant cet exemple dans le formalisme décrit dans la définition 1.1, on obtient :

$A = \{$*Prendre bouilloire, Remplir d'eau, Prendre thé, Prendre tasse, Remplir tasse, Prendre lait, Prendre chocolat*$\}$

$NT = \{$*Faire eau chaude, Faire du thé, Faire du chocolat*$\}$

$G = \{$*Faire du thé, Faire du chocolat*$\}$

$P = \{\{$*Faire du thé* $\rightarrow$ *Faire eau chaude, Prendre thé, Prendre tasse, Remplir tasse,* $\sigma = \{(1,4),(2,4),(3,4)\}\}$,

$\{\{$*Faire du chocolat* $\rightarrow$ *Prendre chocolat, Prendre tasse, Prendre lait, Remplir tasse,* $\sigma = \{(1,4),(2,4),(3,4)\}\}$,

$\{\{$*Faire eau chaude* $\rightarrow$ *Prendre bouilloire, Remplir d'eau,* $\sigma = \{(1,2)\}\}\}$

1.1.2 Modèle de décision

Le modèle de décision (*Decision Model*) représente les préférences d'un agent sur l'exécution de certaines règles par rapport à d'autres. Ces préférences sur les règles permettent de pondérer la reconnaissance de plan en utilisant des probabilités. Généralement, celles-ci fonctionnent comme des probabilités à priori sur les objectifs et les sous-objectifs, permettant d'utiliser une approche bayésienne du problème.

Plusieurs chercheurs ont décrit le modèle de décision. Les auteurs de SLIM (*Semi-Lazy Inference Mechanism*),une approche de reconnaissance de plan utilisant des plans hiérarchiques partiellement ordonnés, ont proposé une forme de modèle de décision intégrée à la définition des règles(Mirsky et Gal, 2016). Étant donné que la bibliothèque de plans présentée en définition 1.1 décrit une méthode afin d'atteindre un objectif, indépendamment de l'agent et que le modèle de décision décrit les préférences d'un agent unique, nous proposons de séparer les deux définitions.

Définition 1.2. *Un Modèle de Décision DM associé à une bibliothèque de plans PL est une fonction de distribution de probabilités sous la forme* $DM : NT \times P \rightarrow [0,1]$, *avec les contraintes* $\forall nt \in NT, \sum_{p \in P} DM(nt,p) = 1$.

Dans cette définition, nous décrivons un modèle de décision comme fonction de distribution de probabilités sur l'ensemble des règles de productions d'une biblio-

thèque de plans.

1.1.3 Algorithmes de résolution

Un problème de reconnaissance de plan est une combinaison d'un ensemble d'observations O et d'une bibliothèque de plans PL accompagnée de son modèle de décision DM. Le but est de trouver le plan, c'est-à-dire l'arbre d'actions hiérarchiques partiellement ordonnées, expliquant au mieux les observations. Dans le cas de la reconnaissance d'objectifs, c'est l'objectif le plus probable, c'est-à-dire dans notre cas, la racine des arbres, que l'on va inférer. Une variante commune est de chercher non pas uniquement le plan le plus probable, mais d'inférer la distribution de probabilités des plans.

Definition 1.3. *Soit E l'ensemble des arbres issus d'une bibliothèque de plans PL. Soit O une séquence d'actions $o_1, o_2, o_3, ..o_i \in A$. La solution S à un problème de reconnaissance de plan PR sur (PL, DM, O) est $S = \max_{e \in E}(P(e|O))$.*

Definition 1.4. *Soit E l'ensemble des arbres issus d'une bibliothèque de plans PL. Soit O une séquence d'actions $o_1, o_2, o_3, ..o_i \in A$. La solution S à un problème de reconnaissance d'objectif GR sur (PL, DM, O) est $S = \max_{g \in G}(P(g|O))$.*

Comme dit précédemment, la résolution d'un problème de reconnaissance de plan utilisant ce formalisme se fait en combinant une génération d'hypothèses valides associées à un raisonnement bayésien. Nous commencerons par présenter le raisonnement bayésien avant de nous concentrer sur la génération d'hypothèses de plans.

En utilisant la définition 1.4 et l'application de la règle de Bayes, on obtient

l'équation (1.1).

$$P(e|O) = \frac{P(e)P(O|e)}{P(O)} = \frac{P(e)P(O|e)}{\sum_{e_i \in E} P(e_i)P(O|e_i)} \tag{1.1}$$

Dans l'équation (1.1), $P(e)$ est la probabilité à priori d'un plan, c'est-à-dire un arbre partiellement ordonné d'actions, sachant une bibliothèque de plans et un modèle de décision donné. $P(O|e)$ est la probabilité d'observer une séquence d'actions particulière sachant un plan e.

O étant une séquence d'observations $O = [o_1, o_2, ..., o_n]$, $P(O|e)$ peut se ré-écrire de la manière suivante :

$$P(O|e) = P(o_1|e)P(o_2|e, o_1)...P(o_n|e, o_1...o_n) \tag{1.2}$$

Pour simplifier cette écriture, Geib et Goldman ont proposé deux hypothèses. La première hypothèse est que les observations sont indépendantes les unes des autres, celles-ci ne dépendant que de l'hypothèse de plan e sous-jacente ainsi que de la temporalité de l'observation (comprendre du numéro de l'observation dans la séquence). Ces deux hypothèses ont influencé d'autres auteurs dont (Kabanza *et al.*, 2013) et (Mirsky et Gal, 2016).

En utilisant cette hypothèse, l'équation 1.2 peut s'écrire

$$P(O|e) = P(o_1|e)P(o_2|e)...P(o_n|e) = \prod_{i=1}^{n} P(o_i|e) \tag{1.3}$$

La seconde hypothèse porte sur la distribution $P(o_i|e)$. Cette distribution étant inconnue à priori, Geib et Goldman proposent d'utiliser une distribution uniforme sur l'ensemble des actions possibles de e à l'instant i. Nous appellerons cet ensemble $PS(i, e)$.

$$P(o_i|e) = \frac{1}{|PS(i, e)|} \tag{1.4}$$

Comme pour réaliser un objectif $g \in G$ donné, il faut exécuter un plan e, la probabilité de g peut être obtenue de la manière suivante :

$$P(g) = \sum_{e_i \in E | g \in e} P(e_i) \tag{1.5}$$

Par conséquent :

$$P(g|O) = \sum_{e_i \in E | g \in e} P(e_i|O) \tag{1.6}$$

Ce raisonnement peut être généralisé au cas des plans multi-objectifs de deux manières. La première est d'utiliser des règles de production récursives sur les objectifs afin de permettre l'expression de plusieurs objectifs. C'est la solution que nous utilisons dans nos systèmes, car celle-ci s'adapte bien aux filtres à particules, la méthode de résolution que nous proposons à la section 1.4. La seconde façon de faire est de considérer qu'au cours de l'exécution d'un plan e, plusieurs objectifs $g_i \in G$ peuvent être suivis de manière concurrente. C'est l'approche choisie par Geib et Goldman, car mieux adapté aux approches par dénombrement explicite.

Dans le cas multi-objectifs, la probabilité d'avoir un ensemble d'objectifs *goals* peut être obtenue en utilisant une distribution géométrique si l'on fait les deux hypothèses suivantes : (1) les objectifs sont indépendants entre eux, (2) les plans de ces objectifs ne partagent pas leurs actions, c'est-à-dire que si une action se trouve dans un plan d'un objectif et une autre action de même nature se trouve dans un autre plan, alors les deux actions sont à traiter comme deux actions distinctes.

On obtient alors l'équation suivante :

$$P(goals) = \prod_{g_i \in goals} P(g_i)^n (1 - P(g_i)) \prod_{g_i \notin goals} (1 - P(g_i)) \tag{1.7}$$

Pouvant être simplifiée par :

$$P(goals) = \prod_{g_i \in goals} P(g_i)^n \prod_{g_i \notin G} (1 - P(g_i)) \tag{1.8}$$

Si l'on en revient au plan e, dans le cas multi-objectifs celui-ci peut être décomposé en un ensemble de plans $plans$, pouvant être interprété comme une forêt d'arbres ET/OU. On peut ré-écrire $P(e)$ de la manière suivante :

$$P(e) = P(goals)P(plans|goals) \tag{1.9}$$

Ou $P(plans|goals)$ correspond à la probabilité d'obtenir un certain ensemble de plans $plans$ sachant un ensemble d'objectifs $goals$. Cette probabilité peut être obtenue à la construction des plans en utilisant le modèle de décision.

En utilisant les équations (1.1), (1.7) et (1.9), on obtient l'équation suivante :

$$P(e \wedge O) = \prod_{g_i \notin G} (1 - P(g_i)) \prod_{g_i \in e} P(g_i) \prod_{plans \in e} P(plans|goals) \prod_{i}^{n} \frac{1}{|PS(i,e)|} \tag{1.10}$$

Pouvant être utilisée avec l'équation (1.6) afin d'obtenir une formule finale de $P(g|O)$ dans le cas de la reconnaissance d'objectifs sachant plusieurs objectifs :

$$P(g|O) = \frac{\sum_{e_i \in E|g \in e} P(e_i \wedge O)}{\sum_{e_j \in E} P(e_j \wedge O)} \tag{1.11}$$

La plupart des approches utilisant ce formalisme sont des approches par dénombrement. En utilisant les équations présentées précédemment, on peut retrouver la probabilité de chaque hypothèse à la condition de générer l'ensemble des hypothèses de plans valides. Une hypothèse de plans valide est une hypothèse supportée par les observations. Il s'agit d'un raisonnement assimilé à un raisonnement abductif. Dans le cas d'un grand nombre d'hypothèses, on peut générer

un sous-ensemble significatif d'hypothèses pour estimer ce dénombrement et les probabilités des plans. Cette méthode est plus rapide, mais approximative. Cette génération d'hypothèses est lourde en calcul. Ces approches sont au moins NP-difficiles dans le cas d'une observabilité parfaite, c'est-à-dire que l'ensemble des actions exécutées par l'agent observé sont connues (Vilain, 1991) et le problème devient semi-décidable dans le cas général (Behnke *et al.*, 2015).

PHATT

PHATT est le nom des algorithmes proposés par Geib et Goldman pour réduire le problème de reconnaissance de plan en utilisant le formalisme présenté dans ce chapitre. Ces algorithmes utilisent le raisonnement bayésien présenté précédemment et une des deux méthodes de production des hypothèses de plans suivantes, la méthode descendante *(top-down)* ou la méthode ascendante *(bottom-up)*.

L'approche *(top-down)* consiste à générer toutes les hypothèses de plans en partant des racines des arbres de plans. Pour chaque racine on choisit une règle de production associée à cette racine et on étend l'arbre ET/OU d'un palier, puis on recommence à partir d'un des éléments de ce nouveau palier, on choisit une règle et on descend jusqu'à obtenir une feuille. Si cette feuille correspond à l'observation attendue, l'hypothèse est valide et on calcule sa probabilité. Pour chaque nouvelle observation, on prend toutes les hypothèses valides à l'observation précédente et on recommence.

Cette méthode de génération est exhaustive, ce qui induit qu'elle génère un certain nombre d'hypothèses intermédiaires non valides, celles-ci ne pouvant pas expliquer les observations. De plus, dans le cas multi-objectifs, le nombre d'hypothèses très grand, voire infini, étant donné que n'importe quelle séquence d'objectifs, aussi improbable soit-elle, doit être générée afin de maintenir la validité de l'ensemble

d'hypothèses à dénombrer.

Pour pallier ces problèmes Geib et Goldman proposent l'approche *bottom-up*. Celle-ci exploite les propriétés abductives de la reconnaissance de plan en construisant les hypothèses à partir des feuilles et en remontant les arbres jusqu'à leurs racines. Ainsi les hypothèses sont construites en utilisant les observations, ce qui induit seules les hypothèses qui sont valides au moment de l'observation sont construites, contrairement à l'approche *top-down*.

De manière simplifiée, l'algorithme fonctionne de la manière suivante : pour chaque observation on choisit une règle de production qui contient l'observation comme un des éléments de sortie. Ainsi le symbole non terminal d'origine de cette règle est une hypothèse valide. On applique ce raisonnement en prenant cette fois ce symbole non terminal à la place de l'observation. On remonte ainsi l'arbre jusqu'aux racines. Toutes les hypothèses recouvrant toutes les observations sont considérées valides et on calcule leurs probabilités.

Pour la construction d'une hypothèse unique, l'approche *bottom-up* est moins rapide que l'approche *top-down* mais comme le nombre d'hypothèses à maintenir est plus faible, le coût pour l'ensemble est plus faible.

Cette approche reste cependant lente même pour des problèmes de taille réduite (Kabanza *et al.*, 2013). La raison principale est que le nombre d'hypothèses à générer est exponentiel à la fois sur la taille de bibliothèque de plans, c'est-à-dire le nombre de règles, d'objectifs et d'actions de bas niveau, et sur la taille des plans en eux-mêmes.

DOPLAR

DOPLAR (*Decision-Oriented PLAn Recognizer*) est l'approche proposée par Kabanza et al. pour traiter en temps réel les problèmes de reconnaissance de plan et pallier aux problèmes posés par l'aspect exponentiel de l'approche Geib et Golman (Kabanza *et al.*, 2013). DOPLAR utilise un raisonnement très similaire à celui de l'approche *bottom-up* de PHATT, mais plutôt que de calculer la probabilité des hypothèses de plan générées, les auteurs proposent de calculer le poids d'hypothèses partielles. Une hypothèse partielle est une hypothèse qui n'a pas été complètement développée. En s'intéressant aux hypothèses partielles, plutôt qu'aux hypothèses, on limite le nombre d'hypothèses à générer. Ceci vient avec un coût, cette approche n'est pas une approche exacte, mais approximative.

Kabanza et al. définissent le poids d'une hypothèse de la manière suivante :

$$poids(e \wedge O) = \sum_{e_i \in DH(e,O)} P(e_i \wedge O) \tag{1.12}$$

Où $DH(e, O)$ correspond à l'ensemble des explications parentes de e, c'est-à-dire partageant des sous-objectifs en commun avec e.

Kabanza et al. propose ensuite le lemme suivant pour obtenir une limite supérieure au poids de (e, O) :

$$ub(poids(e \wedge O)) = P(e \wedge O_{[0:i]}) \prod_{k=i+1}^{n} (1 + \sum_{g \in I | o_k \in PS(k,g)} P(g)) \tag{1.13}$$

En utilisant ce lemme et l'équation (1.10) on peut obtenir deux encadrements, inférieur et supérieur, permettant de générer un intervalle de confiance sur la probabilité de chaque objectif. Pour obtenir une prédiction, on calcule et développe les hypothèses par ordre décroissant de poids jusqu'à obtenir un indice de confiance

compris dans un intervalle prédéterminé.

DOPLAR est un bon candidat pour une application de reconnaissance de plan temps réel embarquée sur un robot assistant. Cependant cette approche est limitée dans sa gestion des observations imparfaites (voir section 1.2) qui sont gérées ici de la même manière que pour PHATT.

Réseau de neurones récursif

Bisson et al. ont proposé d'utiliser des réseaux de neurones à récursion (RNN pour *Recursive Neural Network*) pour apprendre le modèle de décision et exploiter le réseau obtenu comme un système de reconnaissance de plans (Bisson *et al.*, 2015). Le RNN utilisé par les auteurs représente les structures internes du HTN représentant la bibliothèque de plans en représentant chaque étage en partant des racines par un vecteur de caractéristiques.

Cette approche donne de meilleurs résultats sur la plupart des ensembles de données usuels que d'autres approches par bibliothèques de plans hiérarchiques et des techniques issues de la planification comme méthode de reconnaissance de plan. Cependant elle souffre de deux limitations importantes : (1) l'approche a besoin d'un grand nombre d'exemples, (2) elle ne permet pas de gérer les observations imparfaites.

Il existe de nombreuses autres approches, comme Elixir (Geib, 2009) qui propose d'utiliser une variante des grammaires non contextuelles, les grammaires combinatoires catégorielles (CCG pour *Combinatory Categorial Grammar*), Yappr (Geib *et al.*, 2008) qui propose de traiter le problème comme un problème de ré-écriture de chaînes de caractères ou SLIM (Mirsky et Gal, 2016) qui propose une approche fainéante combinant les algorithmes *top-down* et *bottom-up*.

1.2 Observabilité imparfaite

Dans la plupart des applications comprenant des observations, notamment en environnement réel, certaines de ces observations peuvent contenir des informations erronées ou incomplètes. Ces observations imparfaites sont dites alors bruitées (*Noisy observations*). En reconnaissance de plan, en utilisant le formalisme décrit en section 1.1.1, on parle d'observations imparfaites, ou de bruit, lorsqu'une séquence d'observations $O = o_1, o_2, ..., o_n$ faite sur un agent observé ne correspond pas à la suite d'actions $Act = a_1, a_2, ..., a_m$.

On peut identifier trois types d'observations imparfaites qui peuvent survenir en reconnaissance de plan (Vattam et Aha, 2015a) :

- Les observations manquantes (*missing observation*), lorsqu'une des actions n'a pas été observée et est donc absente de la séquence O ;
- Les observations erronées (*mislabeled observation*), lorsqu'une observation o_i sur a_i a bien été faite, que $a_i \in Act$, mais que l'observation ne correspond pas à l'action ;
- Les observations excédentaires (*extraneous actions*), lorsqu' une observation o_i est faite sur une action $a_i \notin Act$.

La plupart des approches décrites précédemment peuvent théoriquement supporter certains types d'observations imparfaites, mais généralement à un coût en calculs élevé les rendant inutilisables dans des cas applicatifs.

Dans le cas des approches issues de PHATT, une solution pour prendre en considération les observations imparfaites est de le modéliser sous forme de règles de production à rajouter à la bibliothèque de plans. Cette approche permet en principe de modéliser chacun des 3 types d'observations imparfaites, mais a comme défaut de rajouter au moins un degré de profondeur à la bibliothèque de plans, ce qui rend le dénombrement des hypothèses beaucoup plus coûteux. Cette modéli-

sation des observations imparfaites peut être appliquée aux approches *top-down*, *bottom-up*, DOPLAR, Elixir, Yappr et SLIM présentées plus tôt.

À notre connaissance, l'approche par RNN ne permet pas une implémentation efficace d'un mécanisme de gestion des observations imparfaites, de l'analyse même des auteurs de cette approche (Bisson *et al.*, 2015).

Dans la plupart des techniques se reposant sur du dénombrement, la gestion des observations imparfaites en reconnaissance de plan se fait par le maintien d'un ensemble d'hypothèses invalides qui auraient été éliminées dans le contexte des observations parfaites, alourdissant les calculs.

Une des autres approches de gestion des observations imparfaites que l'on retrouve dans la littérature consiste à utiliser des métriques de similitude entre les observations et un ensemble d'hypothèses de plans précalculées. Cette utilisation de métrique de similitude se retrouve chez Vattam et Aha ((Vattam et Aha, 2015a)). Dans leurs travaux, les auteurs comparent la séquence d'observations à une bibliothèque de graphes de séquences d'actions (*Action Sequence Graphs*), cette bibliothèque servant de bibliothèques de plans. Une autre famille d'approches où les métriques de similitude sont utilisées concerne l'utilisation conjointe de planification et de la reconnaissance de plan (Ramırez et Geffner, 2009).

Dans nos travaux, nous proposons d'utiliser la définition suivante comme modèle des observations imparfaites :

Definition 1.5 *Une fonction de bruit N sur une bibliothèque de plans PL est une fonction de probabilités sous la forme $N : A \times (A + \emptyset)^p \rightarrow [0, 1]$, avec les contraintes $\forall a \in A, \sum_{o \in (A+\emptyset)^p} N(a, o) = 1$ et $p \in \mathbb{N}$.*

On décrit ici les observations imparfaites comme une fonction de probabilités associées à des règles de mutation. On reprend en esprit le système de règles liées

aux arbres hiérarchiques et on le rapproche des fonctions d'observations qu'on retrouve notamment en planification, de manière similaire au POMDP (*Partially Observable Markov Decision Process*)(voir chapitre 4). Cette définition est un travail original.

Cette fonction peut engendrer quatre différents types de sorties, correspondant aux 3 types d'observations imparfaites décrits précédemment et au cas de l'observation correcte de l'action :

- $a \rightarrow a \mid a \in A$, le cas où l'observation est correcte.
- $a \rightarrow \emptyset \mid a \in A$, le cas où une observation est manquante.
- $a \rightarrow b \mid a, b \in A$, le cas où l'observation est erronée.
- $a \rightarrow B \mid B \in A^p$, le cas des observations excédentaires.

Par la suite, nous faisons la supposition qu'un modèle d'incertitude sous la forme d'une fonction de bruit est donné. Il est important de noter que bien que cette description du bruit est à la fois simple à exprimer et simple à utiliser, la création de cette fonction de bruit peut demander des efforts. La construction de cette fonction ne fait pas partie de cette thèse, même si le problème est abordé en discussion en section 1.6. Cependant nous montrons plus tard dans ce chapitre et dans le chapitre 2 qu'une estimation de la fonction de bruit est suffisante pour obtenir des résultats exploitables, bien qu'ayant un impact sur la précision.

Le choix de définir les observations imparfaites en dehors de la bibliothèque de plans vient du fait que la bibliothèque de plans décrit les étapes à réaliser pour réaliser un objectif, indépendamment de l'acteur ou de l'observateur. Par conséquent, celle-ci est par nature différente du bruit.

Afin de combiner les trois parties que sont la bibliothèque de plans, le modèle de décision et la fonction de bruit, nous introduisons la notion de bibliothèque de plans étendue (EPL pour *Extended Plan Library*).

Definition 1.6 *Une bibliothèque de plans étendue est un tuple* $EPL = (PL, DM, N)$ *où :*

- PL *est une bibliothèque de plans ;*
- DM *est un modèle de décision de* PL *;*
- N *est une fonction de bruit de* PL*.*

1.3 Approches par filtre à particules

Les filtres à particules, aussi nommés filtres particulaires (*particle filter*) (Del Moral, 1997) ou encore Monte-Carlo séquentiel (*Sequential Monte-Carlo*)(Liu et Chen, 1998), sont une classe d'algorithmes d'échantillonnage utilisés pour obtenir une évaluation des probabilités pour un processus Markovien. Le lecteur désirant une bonne introduction au sujet peut consulter (Cappé *et al.*, 2007). Il s'agit d'une famille d'approches approximatives, temps réel, capables de produire des résultats rapidement, mais sans garantie d'obtenir la bonne solution. Cette famille d'approche a des liens avec la famille des approches évolutives, dont les algorithmes génétiques, une autre famille d'approches par population, dans le fait que comme celle-ci, elle nécessite la création, le maintien et le filtrage d'une population. La principale différence est l'absence d'opération de croisement dans les filtres à particules.

Les filtres à particules sont de bons candidats pour la reconnaissance de plan et ce pour plusieurs raisons :

1. ils sont efficaces pour obtenir des distributions de probabilités pour un processus Markovien, or l'exécution de plans comme ceux obtenus par un arbre ET/OU peuvent être vus comme des processus Markoviens car chaque action à l'instant t ne dépend que de l'état de l'arbre à $t-1$;
2. ils sont temps réel, une des problématiques que nous cherchons à aborder pour la reconnaissance de plan ;

3. ils sont efficaces sur des signaux bruités (Bui, 2003), une autre problématique que nous cherchons à aborder.

D'un point de vue formel, si l'on considère le processus Markovien suivant :

$$X_t|X_{t-1} = x_t \sim P(x_t|x_{t-1}) \tag{1.14}$$

ainsi que la fonction d'observation suivante :

$$Y_t|X_t = y_t \sim P(y_t|x_t) \tag{1.15}$$

et que l'objectif est d'estimer la fonction de distribution $P(x_t|y_0, ..., y_t)$, alors on peut obtenir un estimateur en générant N variables aléatoires p de X_0 grâce à la formule

$$\hat{P}(x_t|y_0, ..., y_t) = \frac{1}{N}\sum_{i=1}^{N} \delta_{p_k^i}(x_k) \tag{1.16}$$

Où δ est une mesure de Dirac.

On peut identifier quatre étapes à un filtre à particules :

1. **Initialisation.** On crée une population initiale de simulations de X, appelées particules, à l'état X_0.
2. **Transition.** On fait évoluer par la simulation les particules de l'état t à l'état $t + 1$ et on calcule leur sortie y_t.
3. **Filtrage.** On réalise une observation. On retire de la population toutes les particules dont l'état de sortie est différent de l'observation.

4. **Ré-échantillonnage.** On génère de nouvelles particules en choisissant de manière aléatoire des particules survivantes de la population et en les dupliquant. On retourne ensuite à l'étape 2

Comme beaucoup d'algorithmes d'échantillonnage, les filtres à particules gagnent en précision en augmentant le nombre d'échantillons, ici en augmentant le nombre de particules. Sur la plupart des problèmes, la précision maximale tend à atteindre un plateau passé un certain nombre de particules, ce nombre de particules dépendant du problème.

Les filtres à particules ont déjà été utilisés en reconnaissance de plan, notamment par Bui et al. (Bui, 2003). Dans cet exemple d'utilisation, les auteurs proposent d'utiliser le modèle d'inférence de Rao-Blackwell sur les filtres à particules. La différence entre cette approche et celle présentée dans ce chapitre vient essentiellement de la différence dans la description des bibliothèques de plans. Ici il s'agit d'arbre ET/OU, alors que dans le cas de Bui et al. il s'agit de chaînes de Markov cachées abstraites à mémoire (AHMMM, *Abstract Hidden Markov Memory Model*).

1.4 Application aux plans hiérarchiques

L'idée d'utiliser les filtres à particules en reconnaissance de plans vient du fait que bien que le nombre d'hypothèses soit très grand, seul un sous-ensemble relativement restreint de celles-ci a un impact significatif sur l'estimation du plan de l'agent observé. Par conséquent, le but est de générer un nombre limité d'hypothèses tout en limitant le coût de calcul de chacune d'entre elles.

Nous proposons d'utiliser des arbres de plans partiels comme particules. Les arbres de plans partiels peuvent être générés facilement en utilisant une approche *top-*

down, moins coûteuse que l'approche *bottom-up* (Geib et Goldman, 2009).

On considère la résolution d'un problème de reconnaissance de plans sur la bibliothèque de plans étendus $EPL = (PL, DM, N)$

On choisit de représenter une particule p comme un tuple $p = (g, e, ENO)$ avec :

- g est un objectif de la bibliothèque de plans ;
- e est un arbre de plans permettant de réaliser g ;
- ENO est la prochaine observation attendue pour cette particule (ENO pour *Expected Next Observation*). Il s'agit de la dernière action de bas niveau calculée lors de la mise à jour de l'arbre de plans partiels.

Afin de représenter les arbres de plans partiels, nous utilisons la notation suivante pour les noeuds : un *Symbol Node* est un tuple $SN = (s, C, r, f)$ où :

- s est un symbole de $NT \cup A$ de la bibliothèque de plans PL ;
- C est l'ensemble des *Symbol Nodes* enfants de SN. C permet de représenter les arrêtes de l'arbre de plan partiel. Cet ensemble peut-être vide.
- r une règle de production de PL ayant comme prémisse s. r peut être vide comme dans le cas ou $s \in A$
- f un booléen représentant l'état de SN, complété ou non. Un *Symbol Node* est complet si (1) tout les $c \in C$ sont complets, (2) pour tout symbole $s' \in r$ il existe un $SN \in C$ correspondant.

La racine d'un arbre de plan partiel est un *Symbol Node* ayant pour paramètre s un symbole de G.

Dans notre algorithme, nous utilisons les deux fonctions suivantes :

1. **RNC**(P, S). Cette fonction décrit la sélection de manière aléatoire d'un élément de l'ensemble S en utilisant la fonction de distribution P.
2. **VALIDSYM**(r, C). Cette fonction utilise les contraintes de la règle r sur un ensemble C de *Symbol Nodes* afin de retourner un élément c respectant

ces contraintes, ou $\perp$ s'il n'existe pas d'éléments valides. Un enfant valide doit respecter deux conditions : (1) il n'est pas complet ($f = false$), (2) toutes les contraintes d'ordonnancement sont respectées. Ce symbole est choisi de manière non déterministe sur l'ensemble des symboles valides possibles.

Algorithm 1 Expand Tree

1: **function** $Expand(SN = (s, C, r, f), EPL)$
2: **if** $s \in A$ **then**
3: $f \leftarrow True$
4: **return** $RNC(N, s)$
5: **end if**
6: $c \leftarrow VALIDSYM(R, C)$
7: **if** $c \in C$ **then**
8: **return** $Expand(C(c))$
9: **end if**
10: **if** $c = \perp$ **then**
11: **return** $\emptyset$
12: **end if**
13: $C(c) \leftarrow (c, \emptyset, RNC(DM, c), False)$
14: $result \leftarrow Expand(C(c))$
15: **if** $For\ all\ s\ in\ r,\ C(s).f = True$ **then**
16: $f = True$
17: **end if**
18: **return** $result$
19: **end function**

L'algorithme 1 présente notre algorithme permettant d'étendre un plan partiel, c'est-à-dire dans notre cas de rajouter des noeuds afin d'obtenir une nouvelle

feuille dont le symbole est dans A. Il s'agit d'un algorithme récursif. Il existe deux cas de figure : (1) si le symbole de SN est dans A, la fonction renvoie ce symbole bruité à l'aide de la fonction de bruit, et son statut est fixé à complet ($f = True$), (2) si le symbole de SN n'est pas dans A alors la fonction `Expand` est appliquée sur un enfant valide de SN. Si tous les enfants de SN sont complets et s'il n'y a pas d'autre symbole disponible dans r, alors SN change son statut pour complet. Si il n'y a pas de symbole valide, la fonction renvoie un ensemble vide ($\emptyset$).

Cet algorithme est similaire à celui généralement employé pour l'approche *top-down* de PHATT. Nous l'utilisons ici comme un générateur de particules. Il existe cependant deux différences fondamentales entre notre approche et celle proposée par Geib et Goldman : (1) dans notre approche, les arbres sont développés un noeud après l'autre afin de limiter à la fois la taille des arbres générés et l'engagement de chaque particule. On parle d'engagement pour indiquer le degré d'évolution possible qu'une particule peut réaliser dans le futur. (2) Nous exploitons les observations imparfaites directement dans cet algorithme d'expansion. Gérer les observations imparfaites directement dans l'algorithme permet de garder l'expressivité du modèle tout en n'ayant pas à générer plus tard un grand nombre d'hypothèses liées au bruit.

Pour simplifier l'usage de cette fonction et permettre la génération de plans partiels en utilisant le modèle de décision pour choisir les racines, nous ajoutons un symbole faisant office de primitive dans l'ensemble NT du modèle de décision. Ce symbole est accompagné d'une règle de production permettant de générer des objectifs.

En reprenant la figure 1.1 comme exemple, si l'on a deux objectifs distincts $Faire\ du\ the$ et $Faire\ du\ chocolat$, alors nous ajoutons un symbole NT que nous appellerons $Objectifs$. $Objectifs$ est accompagné de deux règles de produc-

tions de type $Objectifs \rightarrow Faire\ du\ the$ and $Objectifs \rightarrow Faire\ du\ chocolat$. La probabilité de ces deux règles décrivent la probabilité à priori de chacun des objectifs.

Ce système peut être utilisé pour gérer le cas multi-objectifs. Au lieu de juste créer des règles ne contenant qu'un seul objectif, une séquence d'objectifs peut être générée. Une seconde façon de faire est de créer des règles de production faisant référence à elles-mêmes si l'on ne connaît pas à l'avance la limite des séquences d'objectifs.

1.4.1 Initialisation

Nous utilisons une structure appelée Pop qui contient toutes les particules. Nous déclarons Pop comme un dictionnaire ayant comme clé les symboles $s \in A$ et un ensemble de particules p comme valeur. s correspond à la prochaine observation attendue tel que produit par l'algorithme 1. Nous utilisons l'algorithme 2 pour générer la population initiale.

Algorithm 2 Initialization

1: **function** $Init(size)$
2: **while** $Size(Pop) < size$ **do**
3: $s \leftarrow RNC(DM, goal)$
4: $T \leftarrow (s, \emptyset, RNC(DM, s), False)$
5: $ENO \leftarrow Expand(T)$
6: $p \leftarrow (s, T, ENO)$
7: $Pop[ENO].insert(p)$
8: **end while**
9: **return** Pop
10: **end function**

$RNC(DM, goal)$ représente le choix aléatoire d'un objectif, ou de plusieurs objectifs, comme racine de l'arbre de plans partiels. Ce choix se fait en utilisant le modèle de décision tel que décrit plus tôt.

Comme toute approche utilisant des filtres à particules, l'objectif est de maintenir une population suffisamment grande pour obtenir un résultat fiable. Généralement plus il y a de particules, plus l'estimation est précise, sachant que dans la plupart des problèmes, il existe une limite au nombre de particules au-delà de laquelle le gain de précision par particule devient négligeable.

1.4.2 Filtrage et ré-échantillonnage

À chaque observation la population est filtrée en utilisant l'algorithme suivant.

Algorithm 3 Filtering algorithm

1: **function** $Filter(Pop, o \in A)$
2: $size \leftarrow Size(Pop)$
3: $FilteredPop \leftarrow Pop[o]$
4: $Clear(Pop)$
5: **while** $Size(newPopulation) < size$ **do**
6: $p \leftarrow random\ select(FilteredPop)$
7: $p.ENO \leftarrow Expand(p.T)$
8: $Pop[ENO] \leftarrow insert(p)$
9: **end while**
10: **return** $ePopulation$
11: **end function**

En pratique, pour pouvoir gérer les actions excédentaires, le ENO est traité comme une file d'actions plutôt que comme un singleton.

Afin de résoudre le problème de reconnaissance d'objectifs, c'est-à-dire de trouver

quel est l'objectif le plus probable réalisé par l'agent observé, il suffit de compter le nombre de particules correspondant à chaque objectif. L'objectif le plus probable est l'objectif ayant le plus de particules.

$$\widehat{P}(G|O) = \frac{1}{|Pop|} \sum_{p \in Pop} \delta_p(p.G = G) \tag{1.17}$$

L'équation (1.17) présente le calcul de l'estimateur $\widehat{P}(G|O)$ où G est un objectif ou un ensemble d'objectifs et δ est un filtre de Dirac.

Avec cette version de l'algorithme, nous réalisons la génération/expansion des particules avant l'étape de filtrage. Cette façon de faire nous permet d'avoir un estimateur de $\widehat{P}(o_{t+1}|o_t)$ à travers le ENO de chaque particule. Cette approche peut être modifiée afin de réaliser les étapes d'expansion et de filtrage au même moment.

En utilisant les algorithmes 1 et 3, nous évitons la génération d'un nombre exponentiel d'hypothèses dépendant des observations. Ici les observations servent uniquement à filtrer et n'ont pas d'impact direct sur la génération des arbres de plan partiels.

La figure 1.2 présente un exemple d'application du filtre à particules sur la bibliothèque de plans présentée dans la figure 1.1. Le problème ici est d'identifier quelle boisson un agent observé est en train de faire : soit un thé soit un chocolat chaud. Nous nous concentrons principalement ici sur le filtrage et non pas sur la génération des ENO. Nous supposons ici aucune observation imparfaite et une probabilité à priori de $1/3$ pour la réalisation du thé et $2/3$ pour la réalisation du chocolat. À l'initialisation, il n'y a aucune observation faite, nous choisissons de manière aléatoire les objectifs, c'est-à-dire les racines associées à chaque particule. À la première transition, les particules sont étendues afin de générer leurs

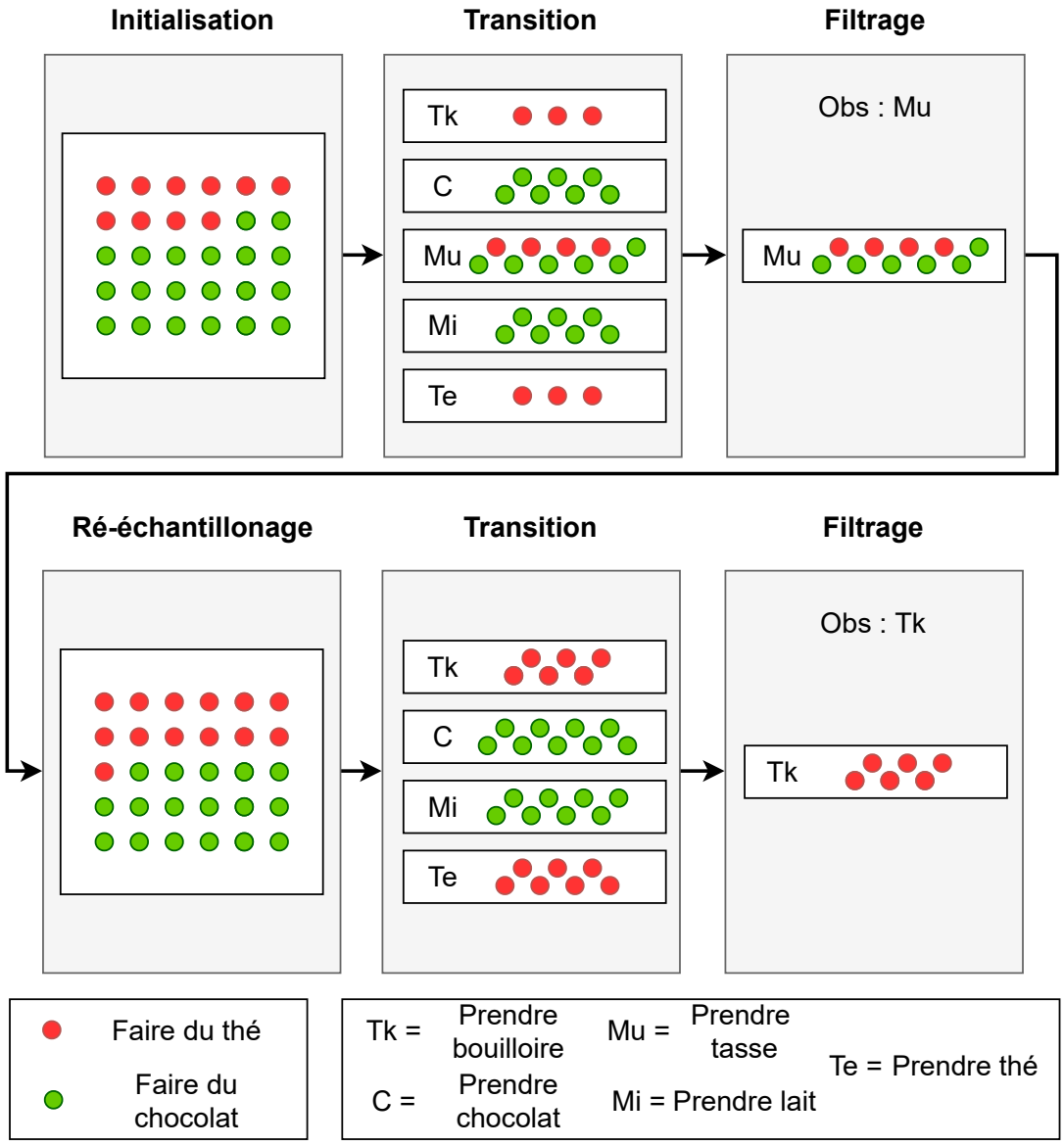


Figure 1.2 Exemple d'application du filtre à particules

premières *ENO*. Dans cet exemple, il existe 5 *ENO* possibles, Tk, C, Mu, Mi et Te. Ces particules sont triées en fonction de leur *ENO*. Ces deux premières étapes correspondent à l'algorithme 2. Quand une observation est faite, ici avec Mu, les particules sont filtrées. Seules les particules dont le *ENO* correspond à l'observation sont conservées. À l'étape de ré-échantillonnage, les particules survivantes sont dupliquées afin de maintenir la taille de la population. S'en suit une nouvelle étape d'expansion pour la seconde transition. Les particules, après être étendues, sont à nouveau triées par leur *ENO*. Ces trois étapes de filtrage, ré-échantillonnage et de transition correspondent à l'algorithme 3. Après le dernier filtrage, seules les particules de *Faire du The* ont survécu, ce qui indique qu'il s'agit de l'objectif derrière les actions de la personne observée.

1.5 Évaluation

Nous avons implémenté cette approche en C++. Nous avons cherché à évaluer cette approche sur trois aspects :

- La précision : le but ici est de savoir si cette approche permet un niveau de précision acceptable par rapport à des approches exactes. Un but secondaire est de trouver, dans des problèmes classiques de reconnaissance de plans, à partir de combien de particules le système obtient un niveau de précision acceptable ;
- l'efficacité : le but est de comparer la vitesse de résolution du filtre à particules par rapport à des approches exactes ;
- La robustesse au bruit : le but est de voir quel est l'impact des observations imparfaites sur l'approche et si celle-ci est applicable dans le cas de données bruitées.

1.5.1 Domaine

Pour les évaluations, nous avons utilisé des bibliothèques de plans identiques à celles utilisées avec DOPLAR (Kabanza *et al.*, 2013) et SLIM (Mirsky et Gal, 2016). Il s'agit de domaines simulés composés de 100 actions de bases, 5 objectifs, une hauteur de 4, des règles d'une longueur de 3 symboles, et un nombre de règles possibles par symbole dans NT de 2. Nous avons considéré l'équiprobabilité sur les règles pour faire le modèle de décision. En ce qui concerne le bruit, pour les tests de précision et d'efficacité, nous avons utilisé la fonction identité. Les fonctions de bruits utilisées dans le cas du test de tolérance seront données plus tard. Nous avons ajouté une probabilité de $1/3$ à l'ordonnancement entre deux paires de symboles dans toutes les règles de productions.

1.5.2 Précision

Afin de tester la précision, nous avons testé notre algorithme sur 100 instances du domaine décrit précédemment en variant le nombre de particules. Pour chaque instance, nous avons généré 10 plans aléatoirement.

La figure 1.3 présente les résultats obtenus. Elle montre en abscisse le pourcentage de complétion du plan de l'agent observé et en ordonnée la précision moyenne de la reconnaissance de plan, c'est-à-dire la moyenne des fois où l'algorithme a renvoyé l'objectif caché. Dans cette figure, nous montrons les résultats obtenus pour 50, 100, 250 et 500 particules. Nous avons aussi expérimenté avec 750 et 1000 particules, mais les résultats obtenus étaient quasiment identiques à ceux obtenus pour 500 particules. Ceci indique qu'au-delà de 500 particules, l'augmentation du nombre total de particules n'améliore pas la précision et qu'on atteint un plateau. Quel que soit le pourcentage de complétion du plan, la précision dépend du nombre de particules, le plus étant le mieux jusqu'à 500.En fonction de la

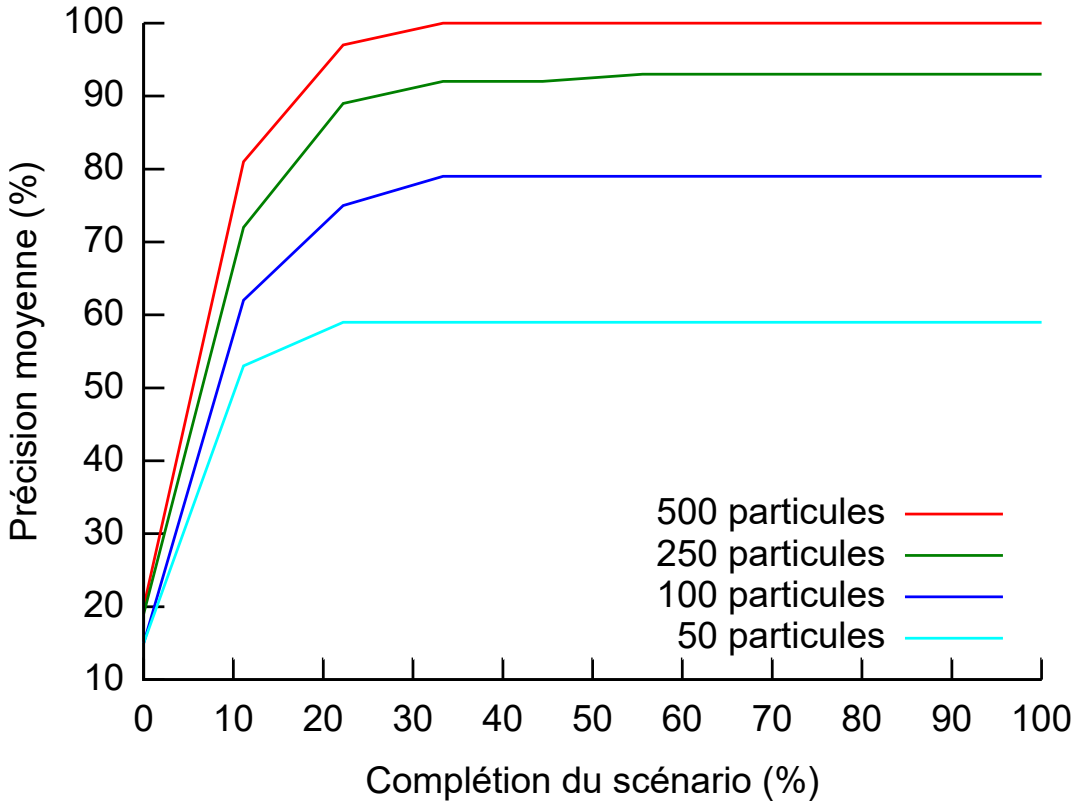


Figure 1.3 Précision de la reconnaissance de plans en fonction de la complétion du plan pour 50, 100, 250 et 500 particules

progression, la précision moyenne augmente d'environ 20% jusqu'à 60% pour 50 particules et de 20% jusqu'à 100% pour 500 particules. Quel que soit le nombre de particules, la reconnaissance de plan atteint un plateau en termes de précision après 30% de la complétion du plan. Ceci indique que l'algorithme fournit une réponse définitive avant la fin de l'exécution du plan. Nous avons comparé ces résultats avec SLIM et PHATT et nous avons observé moins de 1% de différence dans la précision avec notre approche et 500 particules. Ces résultats montrent que cette approche est fiable, même si elle n'est pas complète et ne donne qu'un estimateur des probabilités de chaque objectif.

Ces résultats dépendent des caractéristiques de la bibliothèque de plans. Par exemple, avec plus d'objectifs, comme 10 au lieu de 5, la convergence vers une solution arrive à 20% au lieu de 30%. De même, avec moins d'actions, l'ambiguïté augmente. Avec 10 actions de bas niveau plutôt que 100, la convergence est atteinte vers 60% de complétion et la précision maximale se trouve entre 80% et 90% plutôt qu'à 100%. Nous présentons un exemple de variante de ce domaine avec les résultats associés en chapitre 2.

1.5.3 Rapidité

Nous avons comparé notre approche avec les approches PHATT et SLIM, celles-ci étant des références pour ce type de reconnaissance d'objectifs et ayant une distribution publique de leur code par leurs concepteurs. Il faut noter que le code public des autres approches est en python et notre code en C++, ce qui peut induire un biais dans l'évaluation de l'efficacité.

Tableau 1.1 Temps d'exécution moyen en secondes pour PHATT, SLIM et un filtre à particules de 500 particules (PF-500)

Obs.	1	2	3	4	5	6	7	8	9
PHATT	0.25	0.35	0.57	0.87	1.23	1.71	2.58	4.61	9.88
SLIM	0.17	0.17	0.19	0.26	0.46	0.78	1.28	2.02	4.60
PF-500	0.011	0.019	0.027	0.035	0.043	0.051	0.059	0.067	0.075

Le tableau 1.1 présente le temps d'exécution moyenne pour PHATT, SLIM et un filtre à particules de 500 particules. On peut voir que, comme le nombre de particules à traiter est constant et l'approche est temps réel, la vitesse d'exécution entre 2 observations est constante à 0.008 seconde, à l'exception de la première observation où le temps d'exécution est de 0.011 seconde. Une explication probable serait que ce delta de 0.003 seconde aurait pour origine le temps d'initialisation

des structures de données. Notre approche est significativement plus rapide que PHATT et SLIM dont les temps d'exécution augmentent de manière exponentielle par rapport au nombre d'observations. La différence ne peut être expliquée seulement par la différence de langage C++ versus Python, le fait que notre approche soit temps réel par rapport à SLIM et PHATT qui sont des approches complètes explique en grande partie la différence. Le fait que notre approche ait un temps de calcul faible et surtout constant la rend particulièrement pertinente dans le cas de plans longs, c'est-à-dire avec beaucoup d'observations potentielles, et dans le cas d'applications réelles, souvent avec des contraintes de temps de calcul.

1.5.4 Robustesse aux bruits

Nous avons ensuite évalué la robustesse de notre approche face à des données bruitées. Pour cela, nous avons ajouté une fonction de bruit à nos bibliothèques de plans. Nous avons réalisé les tests avec 3 niveaux de bruits de 10%, 20%, et 30%. Ces pourcentages correspondent aux ratios d'observations imparfaites, quel que soit le type de bruit, c'est-à-dire observation manquante, observation erronée ou observation excédentaire. Pour ces tests, nous avons fixé la même probabilité pour chacun des types d'observations imparfaites. Nous avons fixé les populations de particules à 250, 500 et 1000 pour voir l'impact du nombre de particules sur la précision dans le cas où il y a présence d'observations imparfaites.

La figure 1.4 présente la précision moyenne de la reconnaissance d'objectifs en fonction de la complétion du plan pour 3 tailles de population différentes et trois niveaux de bruit différents. Pour 250 particules dans le cas de l'observabilité parfaite, un plateau dans la précision est atteint à environ 30% de complétion du plan, avec une précision d'environ 95%. Ce plateau est atteint à 30% quel que soit le pourcentage de bruit dans les observations. La précision totale décroît linéairement

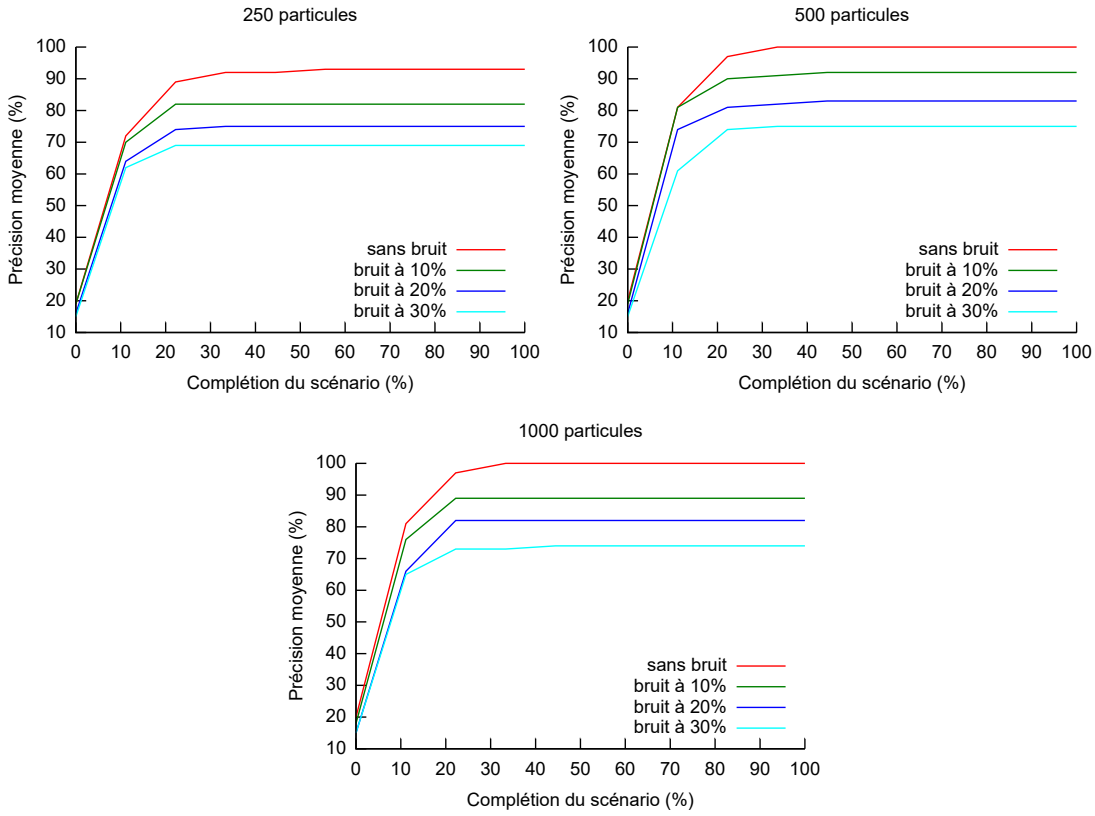


Figure 1.4 Précision de la reconnaissance de plan en fonction de la complétion du plan pour 250, 500 et 1000 particules pour un bruit de 0%, 10%, 20% et 30%

en fonction du pourcentage de bruit, avec un minimum à 65% de précision pour 30% de bruit. Augmenter le nombre de particules à 500 ne change pas le niveau de complétion de plan nécessaire pour atteindre le plateau de précision. Cependant, avec plus de particules, la précision finale est plus grande, ici de 100% dans le cas l'observabilité parfaite, jusqu'à 70% avec 30% de bruit. Augmenter ce nombre à 1000 particules n'augmente pas de manière significative la précision, quel que soit le niveau de bruit choisi. Au vu de ces résultats, un nombre de particules à 500 est suffisant pour obtenir des résultats optimaux à la fois en vitesse et en précision, et ce avec ou sans bruit. Notre approche montre ici une certaine robustesse aux données bruitées, celles-ci étant linéairement corrélées à la précision.

Nous avons essayé de voir s'il existe une différence significative entre les différents types d'observations imparfaites sur la précision de l'approche.

Nous avons testé notre approche avec 20% de bruit avec dans 4 contextes différents : (1) juste des observations manquantes (M), (2) juste des observations erronées (ML), (3) juste des observations excédentaires (E), (4) tous ces cas de figure répartis de manière équiprobable (A).

Tableau 1.2 Précision moyenne finale en pourcentage pour un filtre à particules avec 500 particules.

type de bruit	20% M	20% ML	20% E	20% A
PF-500	83	79	83	81

Le tableau 1.2 présente les résultats obtenus. Les différences obtenues sont suffisamment faibles pour dire que nous n'avons pas trouvé d'impact significatif du type d'observations imparfaites dans la précision finale.

1.6 Discussion

Notre approche offre une réponse à deux problématiques de la reconnaissance de plan avec bibliothèque de plans, la robustesse au bruit et la rapidité. Un point laissé en suspend est la construction de ces bibliothèques de plans. Dans le cas de notre approche où nous utilisons des bibliothèques de plans étendues, les 3 paramètres que sont la bibliothèque de plans, le modèle de décision et la fonction de bruit devraient être construits. Nous discuterons ici de la possibilité d'apprendre ces 3 paramètres.

Le modèle de décision peut être apprit de plusieurs manières. Une approche naïve si l'on a des traces et une bibliothèque de plans consiste à déterminer la fréquence d'utilisation de chaque règle par dénombrement et ainsi en retrouver la fonction de probabilité décrivant le modèle de décision. Une autre possibilité est d'utiliser des RNN ,comme présenté plus tôt dans ce chapitre (Bisson *et al.*, 2015), ou des LSTM (*Long Short-Term Memory*) (Min *et al.*, 2016) .

En ce qui concerne la fonction de bruit, si l'on a une bibliothèque de plans et des traces, une solution naïve consiste à comparer ces traces au plan sous-jacent et estimer la fonction de bruit par dénombrement. Si le plan sous-jacent est inconnu, une façon de procéder serait de chercher les plans valides ayant les mêmes objectifs que les traces et ayant le plus d'actions en commun, dans le même ordre. On considère ces plans valides comme les plans sous-jacents et on applique un dénombrement pour obtenir la fonction de bruit.

Au final, la difficulté principale de la construction de bibliothèque de plans étendu se trouve sur la bibliothèque de plans en elle-même, c'est-à-dire le CFG partiellement ordonné. Nous avons trouvé deux travaux qui semblent particulièrement intéressants pour ce problème : l'utilisation de planification par repères (*land-*

mark) (Pereira *et al.*, 2020a) et l'apprentissage de CCG (Geib et Kantharaju, 2018).

La planification par repères est une technique de planification automatique(Hoffmann *et al.*, 2004). Les repères (*landmark*) sont des éléments intermédiaires qui doivent nécessairement être vrais avant la réalisation d'un objectif. Il peut s'agir d'ensembles actions, de séquences d'actions ou d'états du système.

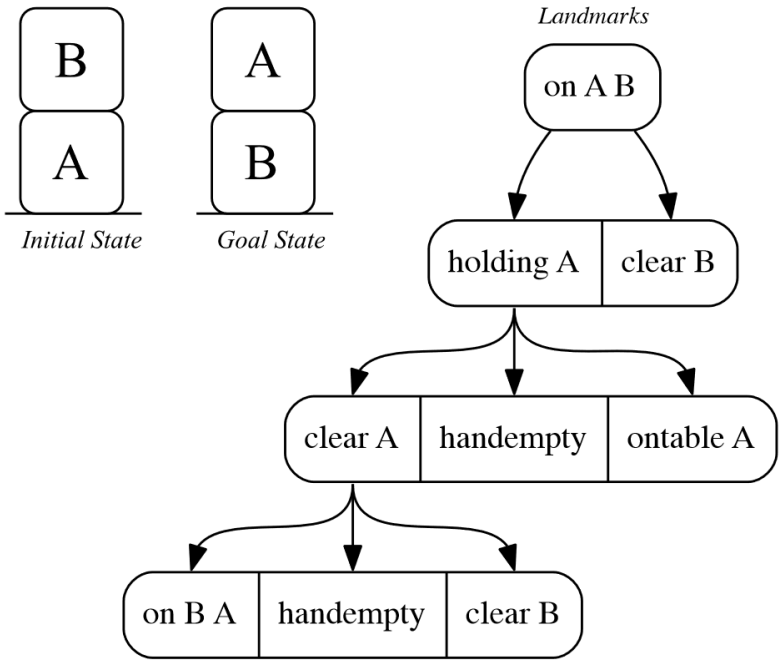


Figure 1.5 Exemple de repères pour la réalisation d'un objectif. Source : (Pereira *et al.*, 2020a)

La figure 1.5 présente les repères pour un problème de planification où l'on cherche à inverser l'empilement de 2 cubes A et B. On remarque une représentation similaire à celui des HTN.

Les repères ont étés utilisés par Pereira et al. pour réaliser de la reconnaissance d'objectif basée sur de la planification inversée (Pereira *et al.*, 2020a). Pour réa-

liser la reconnaissance, les auteurs proposent d'utiliser le taux de complétion des repères, qui sont utilisés alors comme des sous-objectifs d'un plan hiérarchique. Le parallèle entre cette représentation hiérarchique des plans et les bibliothèques de plans que nous présentons ici peut être fait. Une idée pour construire automatiquement des bibliothèques de plans serait donc d'utiliser une génération de repères à partir d'un domaine de planification et d'exprimer ces repères hiérarchisés en utilisant la définition 1.1 de ce chapitre. Une limite de cette idée et que le problème de l'apprentissage est déplacé à l'apprentissage des domaines de planification en eux-mêmes, problème lui aussi ouvert (Arora *et al.*, 2018).

En ce qui concerne l'apprentissage direct des CFG, Geib et Pavan proposent l'algorithme LEX_{learn} (Geib et Kantharaju, 2018) (Kantharaju *et al.*, 2019). Il s'agit d'un algorithme glouton permettant d'apprendre et de lier des séquences d'actions fréquentes et ainsi reconstruire un CCG.

Cet algorithme offre de nombreuses perspectives, cependant celui-ci est limité par plusieurs facteurs : (1) l'approche est peu extensible et le système de reconnaissance de plan associé à cette technique, ELIXIR, a des difficultés à supporter des problèmes de grande taille, (2) l'algorithme ne supporte qu'une forme de bruit dans les traces, les observations manquantes, (3) l'algorithme ne supporte pas les éléments partiellement ordonnés. A noter que le point (1) peut être contourné en utilisant notre approche utilisant des filtres à particules, ceux-ci étant plus rapides et extensibles qu'une approche par dénombrement comme ELIXIR.

1.7 Conclusion

Nous avons présenté dans ce chapitre notre nouvelle approche de reconnaissance de plans utilisant une bibliothèque de plans sous forme d'arbres hiérarchiques. Cette approche est temps réel tout en conservant la richesse d'expression que

donne l'utilisation des grammaires non contextuelles. En plus de ces avancées, nous avons aussi présenté notre définition du bruit qui vient enrichir la définition des bibliothèques de plans. De plus, notre approche à base de filtres à particules permet de supporter ce bruit.

Dans des travaux futurs, il serait pertinent de trouver une méthode pour déterminer à priori un nombre de particules optimal. Cependant comme nous l'avons montré, un nombre restreint de particules donne des résultats très satisfaisant avec un précision de plus de 99%. Un autre problème non traité ici concerne l'apprentissage des bibliothèques de plans. Il faut toutefois noter que ce problème concerne le domaine de manière général et qu'il s'agit d'un problème commun à de nombreux travaux en reconnaissance de plan.

Notre approche répond aux deux grandes limitations des approches antérieures qui limitaient l'utilisation des grammaires non contextuelles dans des applications en environnements réelles. Ceci permet de tester de nouvelles approches pour des systèmes embarqués, notamment des robots assistants où l'utilisation temps réel et le bruit pourront être maintenant supportés.

CHAPITRE II

RECONNAISSANCE DES ACTIVITÉS QUOTIDIENNES

Comme discuté en introduction de cette thèse, notre objectif est de donner la capacité à un robot de reconnaître ce que fait une personne, plus spécifiquement de reconnaître ses ADLs. La reconnaissance des ADLs est un sous-problème issu de la reconnaissance d'activité, notamment des activités humaines. La définition exacte de ce qu'est une activité est une définition floue. En fonction des auteurs, celle-ci peut être décrite de nombreuses manières (Poppe, 2010), cette description dépendant de l'approche de résolution ainsi que de la nature des capteurs utilisés.

Une de ces approches est d'utiliser la reconnaissance de plan comme description des activités de haut niveau et de combiner cette reconnaissance à une reconnaissance d'activité de bas niveau (Bui, 2003) (Granada *et al.*, 2017). Les deux limitations principales de ce type d'approche, souvent qualifié d'hybride, sont : (1) un besoin de précision important de la reconnaissance des activités de bas niveau, avec souvent l'utilisation de capteurs dans l'environnement (2) le besoin de faire un compromis entre l'expressivité du modèle et la rapidité de résolution.

Dans le chapitre précédent, nous avons présenté notre algorithme de reconnaissance de plan ayant une certaine robustesse aux observations imparfaites et permettant une grande rapidité de résolution sans sacrifier l'expressivité.

Notre objectif ici est d'exploiter les propriétés de cette approche de reconnaissance de plan par filtre à particules pour obtenir un système qui soit capable de reconnaître des activités humaines. Cette application, pour pouvoir être utilisée efficacement sur un robot assistant mobile, doit répondre à plusieurs contraintes :

— elle doit être suffisamment rapide pour permettre une interaction entre le robot et la personne observée avant que celle-ci ne fasse une autre activité ;
— elle doit offrir un degré de précision permettant à une application tierce une prise de décision raisonnable ;
— comme nous sommes sur un robot mobile, supposément sans capteur extérieur, elle doit ne nécessiter comme entrée qu'une caméra type RGB-D (*Red Green Blue - Depth*).

Pour réaliser cet objectif nous proposons ici PARC *Plan and Activity Recognition for Robot*, un système combiné de reconnaissance d'activité et d'objectif. PARC fonctionne sur le principe de l'affordance d'objets : les actions de bas niveau sont représentées par une interaction avec un objet, les plans étant des séquences de ces interactions. Cette représentation permet une reconnaissance rapide des activités avec une certaine robustesse aux observations imparfaites et ne nécessitant qu'une caméra RGB-D comme entrée.

Ce chapitre est principalement basé sur notre article présenté à *l'International Conference on Robotics and Automation* (ICRA) (Massardi *et al.*, 2020).

Ce chapitre se compose des sections suivantes : en section 2.1 nous présenterons les deux grandes familles d'approches que l'on retrouve en détection des ADLs, ainsi que leurs liens avec la reconnaissance de plans ; en section 2.2 nous présenterons PARC, le système combiné de reconnaissance de plans et d'activités que nous avons développé en vue de reconnaître les activités quotidiennes ; en section 2.3 nous présenterons une évaluation de notre approche ; en section 2.4 nous ferons une brève discussion de cette approche avant de conclure.

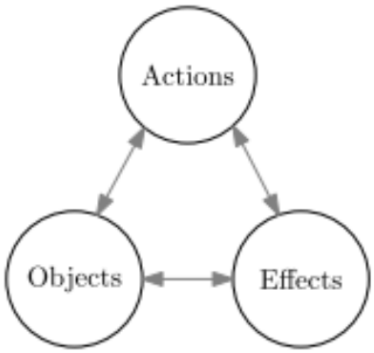
2.1 État de l'art en reconnaissance des ADLs

Il existe de nombreuses approches pour modéliser et reconnaître des actions humaines (Poppe, 2010). On peut cependant se concentrer sur deux grandes catégories : (1) l'utilisation des interactions avec l'environnement et (2) l'utilisation des mouvements et des postures. Ces catégories peuvent être combinées, cependant il faut noter que la plupart des travaux évoqués dans cette section se concentrent sur l'un ou l'autre de ces aspects.

2.1.1 Affordance d'objet

Les interactions avec l'environnent pour caractériser les activités humaines se basent généralement sur le principe d'affordance d'objet (*Object affordance*). Le concept d'affordance désigne à la base une façon de décrire un environnement du point de vue d'un agent (Gibson, 1979). L'affordance d'objet est une spécification où l'environnent manipulable est décrit comme une liste d'objet. En informatique et en robotique, l'affordance d'objet est utilisée pour décrire des activités de bas niveau. Il existe de nombreux exemples d'applications utilisant ce principe. On peut citer par exemple (Montesano *et al.*, 2008) qui représente les actions comme une relation entre des objets, des actions et des relations temporelles, ou (Kjellström *et al.*, 2011) qui simplifie le modèle en n'utilisant que des interactions entre humain et objets. Cette modélisation des activités de bas niveau donne un outil efficace dans le domaine de la reconnaissance d'activités : pour reconnaître une activité il suffit de reconnaître deux éléments, le fait qu'il est une action et l'objet avec lequel l'action est réalisée (Lopes *et al.*, 2007).

L'affordance d'objets est particulièrement utilisée dans le contexte des maisons intelligentes (*smart-home*) dans lesquelles il peut y avoir utilisation de capteurs intégrés dans l'environnement, notamment par l'utilisation de puce RFID (*Radio*



Inputs	Outputs	Function
(O, A)	E	Effect prediction
(O, E)	A	Action recognition/planning
(A, E)	O	Object recognition/selection

Figure 2.1 Représentation des liens d'affordance entre les objets, les actions et leurs effets. Source : (Lopes *et al.*, 2007)

Frequency IDentification) sur des objets (Wu *et al.*, 2007) (Fortin-Simard *et al.*, 2015). Dans le cas de l'étude de Fortin et al., le système présenté obtient une précision de 93.3% pour la reconnaissance d'ADLs. Dans cette étude, les actions sont reconnues lorsque les capteurs posés sur les poignets d'une personne observée s'approchent suffisamment des puces RFID présentent sur les objets.

Bien que ces méthodes donnent un haut niveau de précision, elle nécessite l'utilisation de nombreux capteurs extérieurs ce qui n'est pas compatible dans notre cadre d'application, c'est-à-dire un robot mobile.

Il existe des versions de cette approche se reposant sur des caméras et des caméras RGB-D (*Red Green Blue - Depth*). Ces approches se basent sur une combinaison de reconnaissance d'objets, de pistage d'objets et de détections d'interactions. Avec cette approche les personnes sont le plus souvent représentées comme un

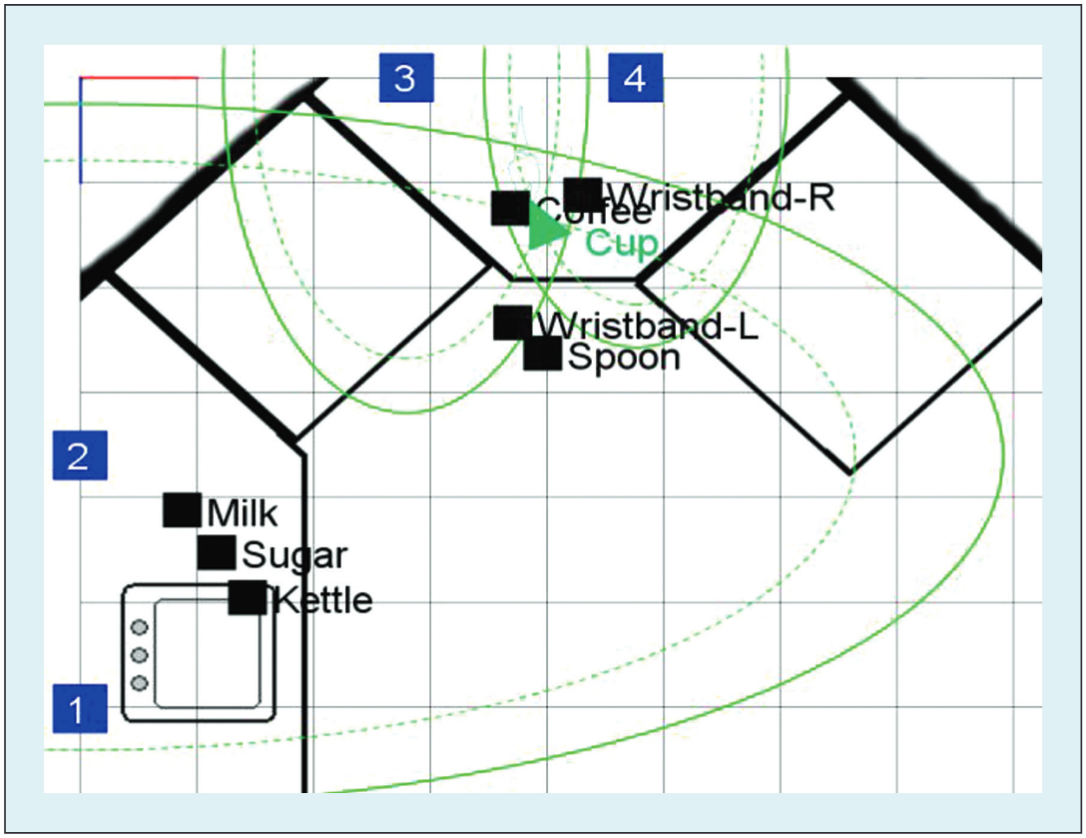


Figure 2.2 Utilisation de puces RFID dans l'environnement et sur les personnes observés. Source : (Fortin-Simard *et al.*, 2015)

des objets à pister (Kjellström *et al.*, 2011) (Song *et al.*, 2013). Une méthode efficace est d'entraîner le système à reconnaître un ensemble d'objets dans le flux vidéo ainsi que les mains de la personne observée. On ajoute ensuite une méthode permettant de déterminer s'il y a contact entre la main est l'objet. En cas de contact, comme pour le cas de capteurs extérieurs, on considère qu'il y a une action en cours. Cette approche peut être étendue pour réaliser de la prédiction d'actions (Dutta et Zielinska, 2018).

Une solution proposée pour reconnaître des ADLs complexes est d'utiliser l'affordance d'objets combinée avec de la reconnaissance de plan ou d'objectifs. Wu et

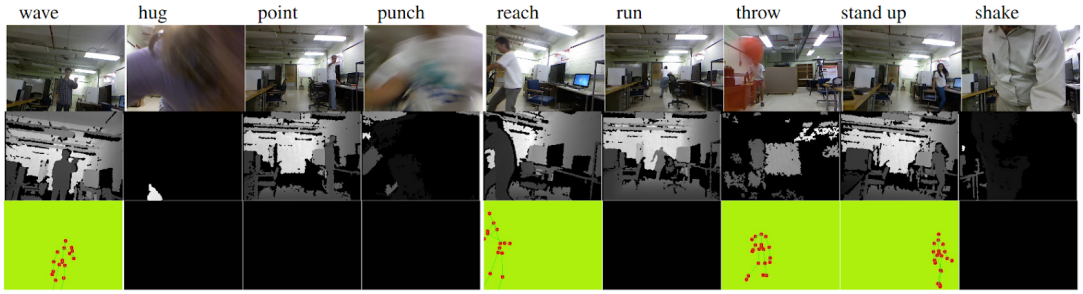


Figure 2.3 Exemple de renaissance d'ADL par analyse de mouvements et de poses. Source : (Xia *et al.*, 2015)

al. utilisent des réseaux bayésiens dynamiques comme méthode de reconnaissance de plan avec l'exploitation de puces RFID comme capteur d'actions de bas niveau (Wu *et al.*, 2007). Rafferty et Nugent utilisent quant à eux une ontologie formelle liée à un système de règles comme système de reconnaissance d'objectifs (Rafferty *et al.*, 2017). En utilisant cette approche, ils obtiennent une reconnaissance quasi parfaite des activités en environnement contrôlé, mais la précision de leur approche descend à 60 % en situation réelle.

2.1.2 Suivi des mouvements

L'autre grande famille d'approche pour caractériser les activités humaines vient de la reconnaissance des mouvements. Celle-ci se fait en règle générale par l'utilisation combinée de capteur RGB-D associé à du suivi du squelette (*skeleton tracking*), l'observant utilisant des éléments visuels comme des estimations de poses ou des mouvements des jointures sur des modèles préentraînés pour reconnaître les activités (Jalal *et al.*, 2015) (Hu *et al.*, 2015) (Xia *et al.*, 2015). En utilisant cette approche et des chaînes de Markov cachées (*Hidden Markov Model*), Xia et al. obtiennent 82% de précision (Xia *et al.*, 2015).

Comme pour les approches utilisant les interactions avec l'environnent, il existe des

approches hybrides combinant reconnaissance de plans et reconnaissance d'activités. Granada et Pereira proposent d'utiliser des images d'une vidéo pour entraîner deux (2) réseaux de neurones à convolution (*convolutional neural networks*), un sur les activités et un second sur les plans (Granada *et al.*, 2017). Avec cette approche, ils obtiennent 67% de précision. Il faut toutefois noter que cette précision est très liée à l'ensemble de données d'entraînement utilisé et décroît rapidement lorsqu'il y a des variations dans le contexte ou dans les vecteurs caractéristiques.

Ce type d'approches par analyse des mouvements souffre de plusieurs limitations. Tout d'abord ces approches se basent sur un apprentissage des activités, ce qui nécessite un ensemble de données important du fait de la grande variance dans la façon de réaliser une action et du fait que l'apprentissage se fait sur un flux vidéo. Cet apprentissage est significativement plus important que l'apprentissage d'objets comme pour les méthodes par affordance. De plus, comme l'analyse se fait sur un flux vidéo, la détection d'activité est plus lente que pour l'affordance et prend le plus souvent quelques secondes. De même, comme il s'agit d'analyser un flux vidéo, cette approche demande plus de ressources, notamment en GPU (*Graphics Processing Unit*), que des approches n'analysant pas un flux vidéo. Enfin une des limitations de cette famille d'approches vient du fait qu'elle modélise souvent les ADLs comme des actions isolées ayant leur propre finalité, comme *se lever*, *lire*, *téléphoner* plutôt que comme une suite d'action ayant une finalité cachée, comme *faire du thé* qui nécessite plusieurs actions de plus bas niveaux pour être réalisée, comme *attraper une tasse* ou *faire chauffer de l'eau*

Dans l'idéal, ces systèmes assistants devraient aussi prendre en compte la probabilité d'actions mal catégorisées, d'observations manquantes, d'occlusions et d'anomalies dans l'exécution des actions. Fahad et al. (Fahad et Rajarajan, 2015) proposent un algorithme qui assigne un score de confiance sur chacune des actions observées et non observées basé sur une analyse temporelle. Cette approche, bien

que prometteuse, ne fonctionne que dans un environnement hors-ligne et non en temps réel.

2.2 Architecture de PARC

Bien que la reconnaissance de plan soit un bon candidat pour la reconnaissance d'actions complexes, on se retrouve en décalage entre ce que les algorithmes de reconnaissance de plans sont en théorie capables d'exprimer et de faire et ce qu'on retrouve en pratique, notamment pour la détection des ADLs. Ce décalage peut s'expliquer par un besoin de rapidité du côté de la reconnaissance de plan pour avoir une application pertinente et de précision du côté de la reconnaissance d'activité, une partie des algorithmes de reconnaissance de plan ayant une forte sensibilité aux observations imparfaites. Dans le chapitre précédent, nous avons introduit un nouvel algorithme de reconnaissance de plan qui répond à ces deux problématiques, étant particulièrement rapide et offrant une certaine robustesse aux observations imparfaites.

Nous proposons d'utiliser notre approche par filtre à particule combinée à de l'affordance d'objet. L'affordance d'objet nous paraît être un bon choix, car : (1) elle se paire assez facilement avec de la reconnaissance de plans (Wu *et al.*, 2007) (Rafferty *et al.*, 2017), (2) elle nécessite moins de ressources qu'une analyse de mouvement, ce qui répond à notre problématique de robot mobile accessible au grand public, (3) elle est en règle générale plus rapide, ce qui donne un avantage pour des applications d'assistances cognitives en temps réel.

Le système que nous avons développé, combinant reconnaissance de plan par filtre à particule et reconnaissance d'activité par affordance d'objets, se nomme PARC pour *Plan and Activity Recognition Component*. PARC a une architecture comme présentée en figure 2.4. Le système fonctionne de la manière suivante : le robot,

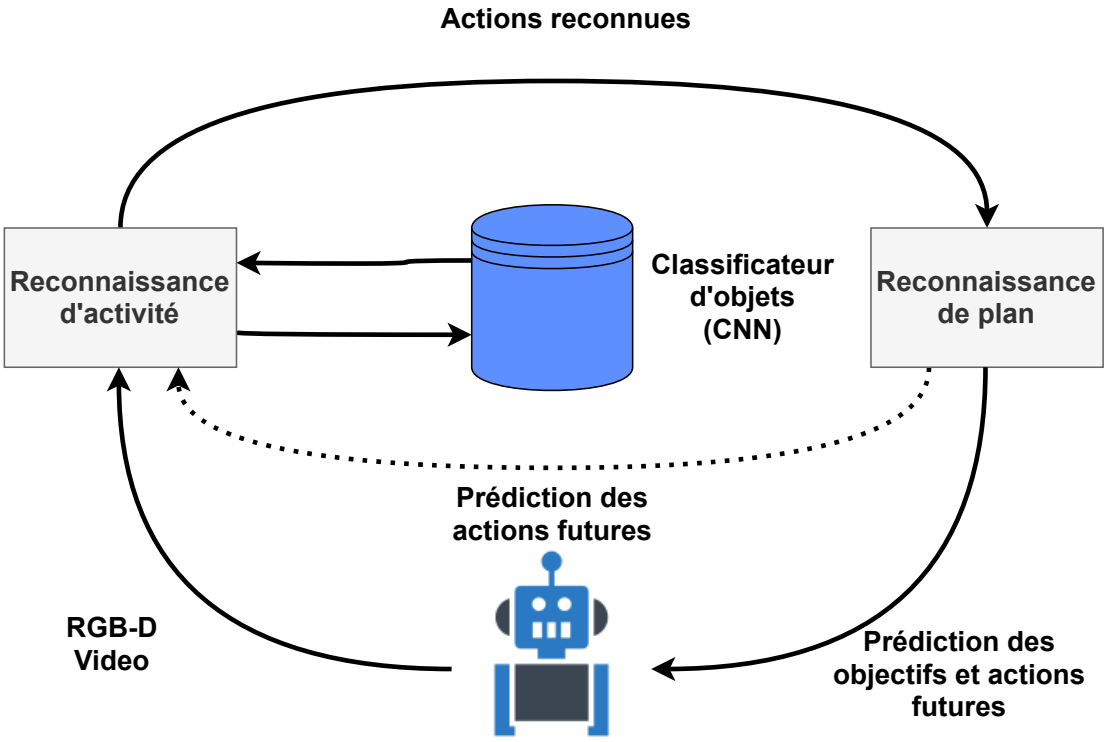


Figure 2.4 Architecture de PARC

par l'intermédiaire d'une caméra RGB-D embarquée, collecte des images et des mesures de profondeur de l'environnement. Ces informations sont envoyées à un module de reconnaissance d'activité par affordance. Ce module détecte si une activité est en cours. Si c'est le cas l'information est envoyée au module de reconnaissance de plan qui calcule quels sont les objectifs les plus probables expliquant les actions observées ainsi qu'une estimation de quelles seront les actions futures.

Étant donné que nous avons déjà présenté la reconnaissance de plan dans le chapitre précédent, nous nous concentrerons ici sur le fonctionnement de la reconnaissance d'activités.

2.2.1 Reconnaissance d'objets et de mains

La reconnaissance d'objets et des mains se fait par le même module, les mains étant traitées comme un type d'objet à part entière. Ce module se base sur un réseau de neurones à convolution pré-entraîné (*Convolution Neural Network, CNN*) construit sur l'architecture de YOLO v3 (Redmon et Farhadi, 2018). Il reçoit du capteur chaque image couleur associée à une carte de la profondeur D. Cette carte de profondeur D est convertie en nuance de gris pour le traitement.

L'image couleur est envoyée à un CNN pré-entraîné permettant une reconnaissance des objets et des mains. Ce CNN réalise cette reconnaissance en faisant de l'analyse de régions. Il peut être exécuté sur CPU en utilisant des techniques d'optimisation (Rajbhandari *et al.*, 2017), ou préférablement sur GPU. Le vecteur en sortie du CNN est sous la forme $Cl_t^o = \{o_t, r_t, p_t\}$ avec o_t la classe de l'objet, r_t la région où l'objet a été reconnu (définie par deux points dans l'espace décrivant un rectangle) et p_t la probabilité associée à la présence de l'objet dans la région. Nous augmentons ces données en exploitant la carte de profondeur D. Pour avoir une profondeur plus précise et plus facilement exploitable sur nos régions, nous utilisons l'algorithme de partitionnement décrit par l'algorithme 4. Celui-ci retire les mesures aberrantes comme montré en figure 2.5. Cet algorithme sépare les données de profondeur pour les faire correspondre aux régions produites en sortie de CNN. Ensuite, celui-ci fait la comparaison entre les valeurs de profondeur de cette région et celle du centre de la région. Si la distance est trop importante alors la valeur de profondeur est retirée et la profondeur moyenne de l'objet est recalculée.

Cette profondeur moyenne d_t des régions est ajoutée au vecteur caractéristique Cl_t^o. Celui-ci est ensuite séparé en deux sous-ensembles, un ensemble H_t pour les mains h_t, et un ensemble O_t pour les objets autres que les mains o_t. Ces deux

ensembles sont ensuite envoyés au module de détection des affordances.

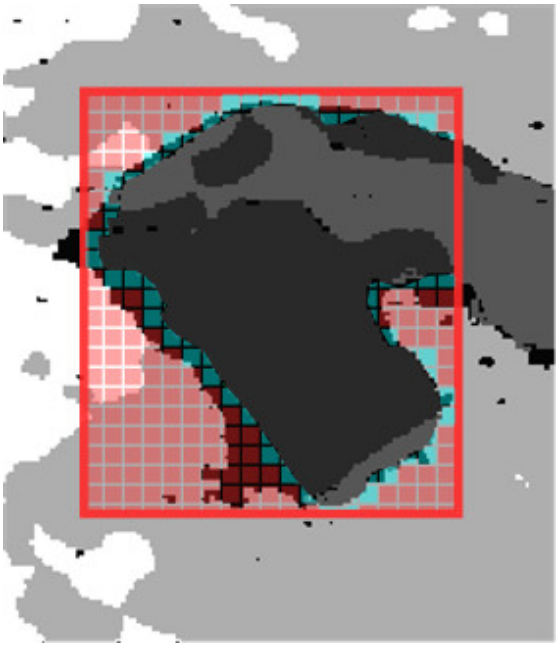


Figure 2.5 Exemple de partitionnement des mesures de profondeurs. Les zones en rouge sont retirées, car jugées aberrantes. Les zones en bleu sont considérées comme ayant une variation acceptable.

2.2.2 Détection des affordances

Comme pour Koppula et Sexena (Koppula et Saxena, 2016), nous discrétisons le temps. Chaque image du flux vidéo est attribuée à un instant t. Pour assurer une bonne efficacité, le delta t séparant deux instants t doit être le plus constant possible, c'est-à-dire la fréquence d'image doit être constante. Pour chaque image, le système reçoit les ensembles ω_H et ω_O, où ω_H contient les régions probables où se trouvent des mains et ω_O contient les régions probables où se trouvent des objets. Ces ensembles sont ajoutés aux ensembles respectifs H_t et O_t.

Le système calcule ensuite la probabilité qu'il y ait une affordance en utilisant

Algorithm 4 Depth mean clustering algorithm

1: **for all** $i \in Cl_t^o$ **do**
2: $C \leftarrow \{c_1, c_2 \ldots_n\} \in r_i \subseteq D$
3: $c_{mil} \leftarrow$ cell in position $\{len(c)/2, len(c)/2\}$
4: $C \leftarrow C \setminus c_{mil}$
5: **for all** $x \in C$ **do**
6: **if** $d(depth(x), depth(c_{mil})) > 0.5$ **then**
▷ d(X,Y) calculates the distance on the depth axis between the two positions X and Y.
▷ If the depth differ too much, we remove the cell.
7: $C \leftarrow C \setminus x$
8: **end if**
9: **end for**
10: **end for**

l'équation suivante :

$$P(OA^t|H^t, O^t) = \prod_{i \in H} \prod_{j \in O} (dist(i,j) distDepth(i,j)) \tag{2.1}$$

Cette équation comprend les éléments suivants :

— OA^t, une affordance entre un humain et un objet ;
— $dist(i,j)$, la distance à l'image (2 dimensions) entre une main i et un objet j. Cette distance est un ratio compris entre 0 et 1, avec 1 la main et l'objet sont superposés ;
— $distDepth(i,j)$, la distance profondeur entre une main i et un objet j. Cette distance est un ratio compris entre 0 et 1, avec 1 la main et l'objet ont la même profondeur moyenne.

Cette équation retourne la probabilité de chaque affordance possible entre un humain et un objet, en fonction des composantes extraites lors de l'observation.

Comme dans notre cas, nous décrivons les actions observables comme une interaction entre un humain et un objet, la reconnaissance d'une action A_t se fait en prenant la valeur de probabilité maximale de $P(OA^t|H^t, O^t)$. Cette valeur est finalement envoyée au module de reconnaissance de plan comme l'observation en cours. Ce module étant une implémentation directe de notre filtre à particules (voir Chapitre 1).

Comme le filtre à particules calcule une prédiction des observations futures, celui-ci peut être utilisé dans une boucle de rétroaction. Le principe est d'utiliser cette prédiction dans la recherche de l'action à observer. Dans ce cas, la détection d'une action A^t se fait en utilisant l'équation suivante :

$$max\{P(A^t|\omega_{OA}) = P(A^t|OA^t, A^{t-1}) \sum_{x=i}^{t-1} P(OA^t|OA^x, H^t, O^t) P(FA^{t-1})\} \quad (2.2)$$

Avec A^t l'action à t et FA^{t-1} la prédiction de l'action par le système de reconnaissance de plan.

Nous avons implémenté et testé cette extension, cependant nous n'avons pas trouvé de différence significative entre l'utilisation et la non-utilisation de la boucle de rétroaction. Nous avons deux explications pour ces résultats : (1) le choix des expérimentations et des domaines pour l'évaluation ne permet pas de mettre en évidence une différence ou (2) la rétroaction n'a effectivement pas ou peu d'impact sur le système. Une étude future plus poussée de l'impact de la boucle de rétroaction pourrait être pertinente, notamment en présence d'ensemble de données plus grand que ceux étudiés ici (voir section 2.3).

2.3 Évaluation

Pour nos expérimentations nous avons utilisé une configuration de matériel similaire à celle utilisée par VIGIL et d'autres robots à bas coût. Nous avons utilisé un TurtleBot [1] doté d'une caméra Intel RealSense D-435, qui est un modèle de caméra RGB-D. Cette caméra était montée sur tripode comme montré en figure 2.6.

Figure 2.6 Turtlebot avec une Intel RealSense D-435 comme prototype pour PARC

PARC a été développé en C++ en utilisant OpenCV comme bibliothèque de traitement d'images et un CNN basé sur YOLO V3(Redmon et Farhadi, 2018).

1. https ://www.turtlebot.com/turtlebot2/

Le CNN est une variation de YOLO V3 où le nombre et la taille des couches ont été réduits pour pouvoir fonctionner en temps réel sur un ordinateur milieu à bas de gamme.

2.3.1 Domaines

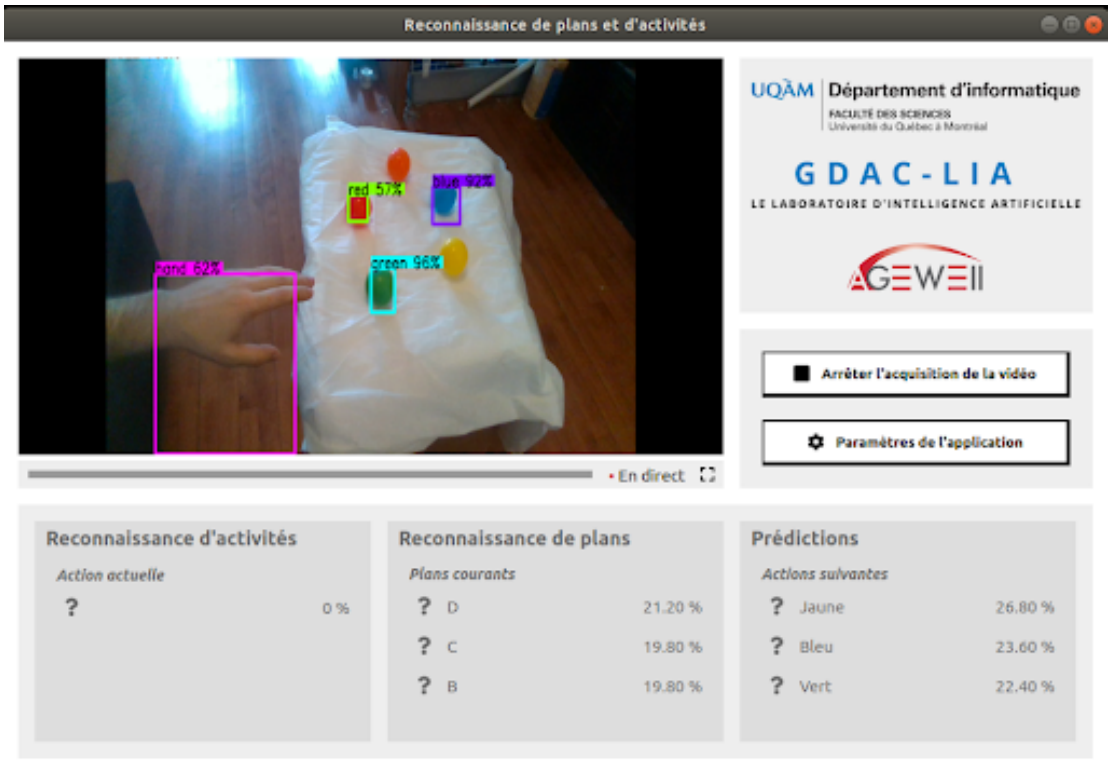


Figure 2.7 Exemple d'une prise de vue de PARC dans les conditions d'expérimentation avec les balles de couleurs

Nous n'avons pas trouvé d'ensemble de données de test public pour la reconnaissance d'activité qui avait à la fois une entrée vidéo et des activités suffisamment complexe pour devoir nécessiter l'utilisation de reconnaissance de plan. La plupart des ensembles de tests pour les ADL se basent sur la reconnaissance d'actions isolées (Jalal *et al.*, 2015) (Hu *et al.*, 2015) (Xia *et al.*, 2015) (Dutta et Zielinska,

2018). L'ensemble de données le plus proche de nos besoins que nous avons trouvés est le Kitchen Scene Context based Gesture Recognition dataset (Shimada *et al.*, 2013) et le CAD120 (Sung *et al.*, 2012), cependant ils souffrent des problèmes suivants pour une expérimentation de PARC : (1) ils ne sont pas structurés comme des problèmes de reconnaissance de plan, (2) ils utilisent du contexte, (3) ils se basent sur la temporalité des actions. PARC ne gère pas pour le moment les points (2) et (3) et un travail long serait nécessaire pour adapter ces ensembles de données pour PARC et ne permettrait de montrer les capacités de notre système à reconnaître les ADLs.

Par conséquent, nous proposons deux ensembles de données qui pourront montrer l'efficacité de PARC : (1) une implémentation directe des domaines décrits en expérimentation dans le chapitre 1 avec un support vidéo et (2) un domaine en preuve de concept de la reconnaissance d'activités en environnement libre.

2.3.2 Intégration entre reconnaissance de plan et reconnaissance d'activité

Pour cette expérimentation, nous nous sommes concentrés sur l'intégration de la reconnaissance de plan avec la reconnaissance d'activité. Nous avons réutilisé les bibliothèques de plans simulés décrites en chapitre 1 en réduisant le nombre d'actions de bas niveau de 100 à 5. Toutes les autres caractéristiques sont identiques, à savoir le nombre d'objectifs de 5, la profondeur de 4, des règles de longueur 3 et 2 règles par symbole dans NT, ainsi qu'un ordonnancement partiel à 33%. Nous avons associé chacune des actions de bas niveau à une balle de couleur pour représenter un objet avec lequel interagir.

Nous avons enregistré 50 vidéos d'exécution de plans issus de ce domaine automatiquement généré dans différents environnements. Pour l'entraînement du CNN, nous avons utilisé des images des mains et des balles prises en dehors des vidéos

d'exécution de plans.

Comme notre approche à filtre à particule nécessite une fonction de bruit, nous avons décidé d'utiliser une surestimation de celui-ci à 40%, avec 30 % d'observations erronées et 10% d'observations manquantes.

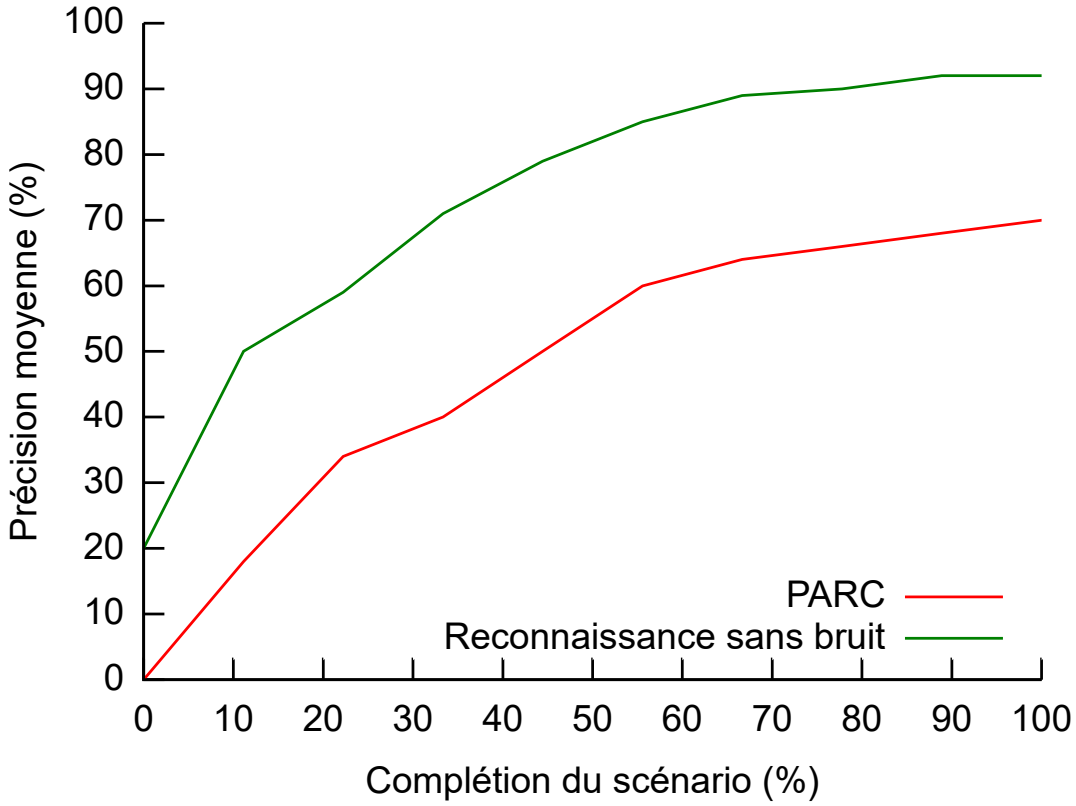


Figure 2.8 Précision en fonction de la complétion du plan

La figure 2.8 présente la précision en fonction de la complétion du plan. En vert la précision dans le cas d'observabilité parfaite, c'est-à-dire la reconnaissance de plan seule sans bruit, et en rouge les résultats obtenus en utilisant PARC.

On peut observer un décalage entre les observations à $t = 0$ et $t = 1$ entre les deux courbes. Ce décalage peut s'expliquer par l'implémentation du filtre à particules :

dans PARC, le système attend une première observation ($t = 1$) avant d'inférer l'objectif caché, alors que dans la reconnaissance de plan seule infère l'objectif sans la première observation.

PARC obtient une précision finale d'environ 70% sur notre ensemble de données. Cette précision est légèrement meilleure que celle attendue qui aurait dû se trouver autour de 65% en utilisant la surévaluation du bruit décrite plus haut. Cette différence restant faible, nous la considérons comme étant dans une marge acceptable. Cette expérimentation montre que PARC permet une intégration efficace entre de la reconnaissance d'activité de bas niveau et la reconnaissance de plan.

2.3.3 Reconnaissance d'ADLs

Comme décrit précédemment, un second ensemble de données a été créé pour tester notre système dans le cas de la reconnaissance d'ADLs. Nous avons créé cet ensemble de données avec deux caractéristiques : (1) il s'agit d'un ensemble de vidéos prises dans un environnement peu contrôlé (2) les actions réalisées sont structurées selon une bibliothèque de plan préétablie.

Cet ensemble se concentre sur les activités liées à la réalisation d'une boisson chaude. Il contient 3 objectifs et 8 actions de bas niveau décrites par le fait de tenir un objet. Les plans sont construits de la façon suivante :

- *Chocomaking(<Hold(teakettle),*
  Hold(Waterpitcher)>,Hold(Mug), Hold(Choco))
- *Chocomaking(Hold(Mug),Hold(Milk), Hold(Choco))*
- *CoffeeMaking(Hold(coffee),Hold(coffeemaker),*
  Hold(water),Hold(coffeemaker),Hold(mug),
  Hold(milk),Hold(mug))

Pour chaque plan, les activités doivent être réalisées dans l'ordre indiqué, sauf

celles présentes entre les marqueurs <>. Celles-ci peuvent être réalisées dans n'importe quel ordre.

Pour construire cet ensemble de données, nous avons demandé à 5 personnes de réaliser les 3 différents plans devant le robot, dans des environnements différents. Pour chaque exécution de plan, nous avons enregistré les données de profondeur et de couleur. Pour une moitié des enregistrements, nous avons déplacé le robot afin de rajouter du mouvement dans la prise de vue. Cet ajout permet de voir si l'approche est robuste quand le contexte dans le flux d'images change pendant la capture, et si le flou dû au déplacement a un impact significatif sur la reconnaissance.

Une vidéo type d'exécution d'un plan décrit par ce domaine et accompagné de commentaires est librement disponible [2] .

Tableau 2.1 Utilisation du processeur et de la mémoire

Type	**Utilisation CPU**	**Utilisation Men.**	**Temps d'éxécution moyen (s)**
Clas. d'objet	**83.7%**	**4.9%**	**0.640**
Reco. d'activité	**0.01%**	**0.3%**	**0.03**
Reco. de plan	**0.08%**	**0.1%**	**0.025**

En moyenne, PARC réussit à reconnaître l'objectif de la personne observée dans 80% des cas. Ces résultats sont significativement inférieurs dans le cas ou l'environnement n'a jamais été utilisé pour l'entraînement de la détection des objets. Dans ce cas, les erreurs de reconnaissance des objets entraînent l'envoi d'observations erronées au système de reconnaissance de plan. Bien que l'approche par filtre à particules offre une certaine robustesse aux observations imparfaites, au-

2. https ://www.youtube.com/watch ?v=QvwBLAb2T58

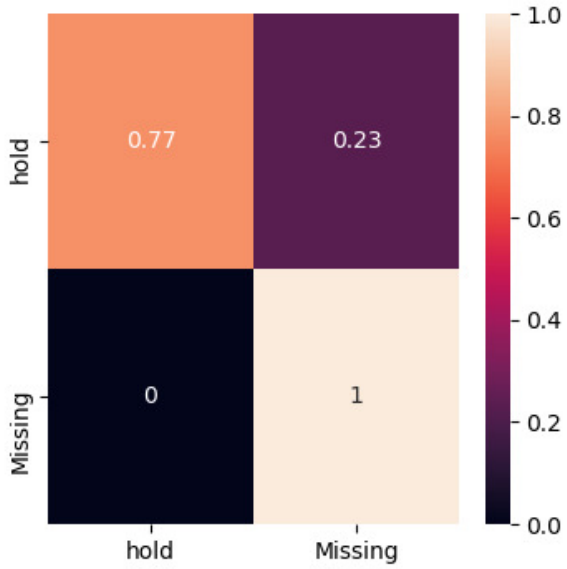


Figure 2.9 Matrice de confusion de la détection d'activité. Missing indique les activités de bas niveau non-reconnu par le système.

delà d'un volume trop important d'observations erronées, le système ne peut plus reconnaître correctement les objectifs.

Le tableau 2.1 présente l'utilisation du processeur et de la mémoire pour les 3 composantes de PARC. Ce tableau montre que la principale contrainte en termes de ressources se trouve dans la classification des objets par notre CNN. Toutes les autres composantes offrent des performances élevées et nécessitent peu de ressources.

2.4 Discussion

Comme indiqué précédemment, nous avons trouvé deux ensembles de données proposant des représentations de problème se rapprochant de la reconnaissance des

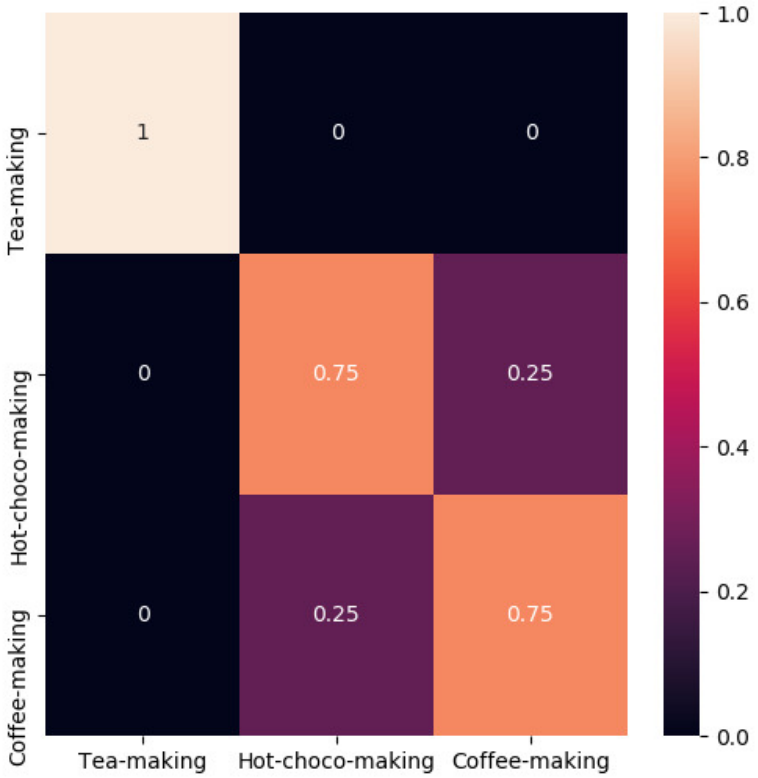


Figure 2.10 Matrice de confusion de la reconnaissance de plan

ADL par reconnaissance combinée des activités et des plans. Ces deux ensembles sont le *Kitchen Scene Context based Gesture Recognition dataset* (Shimada *et al.*, 2013) et le CAD120 (Sung *et al.*, 2012). Un des problèmes soulevés dans l'utilisation de PARC pour ces ensembles vient du fait que dans les deux cas, l'exécution du plan dépend de la temporalité des actions, c'est-à-dire à la fois par la durée des actions et la durée des instants où aucune action n'est en cours.

Pour que PARC puisse adresser la temporalité des actions, celui-ci doit pouvoir supporter les deux points suivants : (1) la durée des actions, (2) la durée des

pauses entres les actions.

En ce qui concerne la durée des actions, une façon de procéder est de définir les actions de la bibliothèque de plans comme des couples actions/durée. Ces durées pouvant être des plages de valeurs plutôt que des valeurs uniques. Un exemple serait de faire des oeufs dur par rapport à faire des oeufs à la coque. La seule différence entre ces deux plans est le temps de cuisson. On peut définir deux actions différentes avec deux plages de temps différentes pour ce temps de cuisson. Dans ce cas, la difficulté porte sur la reconnaissance d'activités qui doit être de discriminer entre plusieurs actions en fonction de leur durée. C'est cette difficulté qui a empêché une implantation sur PARC.

Pour ce qui est des pauses et de leur durée entre les actions, il est possible de définir une pause comme étant une action dans la bibliothèque de plan. Cette action est caractérisée par l'absence d'affordance. Dans ce cas, les observations manquantes sont toutes interprétées comme des pauses, et par conséquent doivent être représentées par la bibliothèque de plans comme des observations erronées ou des observations excédentaires.

Bien que cette extension de PARC soit pertinente pour envisager une application sur le *Kitchen Scene Context based Gesture Recognition dataset* et le CAD120, il existe d'autres limitations telles que l'exploitation du mouvement, les plans en séquences d'actions non hiérarchiques et non ordonnées et le problème de construction de la bibliothèque de plan.

2.5 Conclusion

Dans ce chapitre nous avons présenté PARC, notre système hybride de reconnaissance de plan et d'activité conçu pour fonctionner sur des robots à bas coût dans le contexte de reconnaissance des ADLs. PARC utilise une approche de reconnais-

sance des activités basée sur l'interaction entre humain et objets aussi appelée ici affordance d'objets. L'utilisation conjointe de cette approche et de l'approche de reconnaissance de plan présentée au chapitre 1 permet une reconnaissance rapide des objectifs d'une personne et des actions qu'elle pourrait entreprendre dans le futur.

Une des plus grosses limitations de cette approche provient de la reconnaissance des objets. Il s'agit du module qui prend le plus de temps d'exécution et qui est le moins précis. Dans le cadre de ce projet, nous avions des ressources pour l'entraînement limitées, que ce soit en capacité de calcul ou en nombre de prises de vue des objets dans des environnements divers. Dans ces conditions la reconnaissance d'objets est dépendante du contexte. Ce problème peut être contourné avec plus de puissance de calcul et plus d'images d'entraînements.

Une seconde limitation, à l'impact plus limité, vient du modèle de bruit. Celui-ci nécessite d'être exprimé à priori et devrait être appris. Cependant on note que l'utilisation d'une approximation, dans notre cas une surestimation, est suffisante pour obtenir un modèle exploitable.

PARC pourrait recevoir plusieurs extensions. En plus de celles présentes en discussion, voici trois autres possibilités : (1) Il serait possible d'ajouter la reconnaissance du mouvement en complément de la détection d'affordance. Dans ce cas, le système pourrait reconnaître des activités de bas niveau plus complexes. Cependant cet ajout se ferait par un temps de reconnaissance des activités plus long, or un des avantages de PARC est sa rapidité. (2) Dans l'état actuel du système, chaque affordance est détectée indépendamment de la personne observée, ou de la main interagissant avec un objet. Un système d'identification des agents et d'identification de chaque main, droite et gauche séparément, permettrait une amélioration du système. (3) Avec notre système, la détection d'affordance se fait

par un système de seuil de détection, efficace pour nos tests. l'utilisation de techniques plus avancées pourrait être intéressante dans des cas plus complexes. Un bon candidat serait l'utilisation de LSTM (*Long Short Term Memory*) (Hochreiter et Schmidhuber, 1997).

CHAPITRE III

DÉTECTION D'ANOMALIES DANS L'EXÉCUTION DE PLANS

Pour des systèmes assistants, notamment des assistants cognitifs, il est important de surveiller si une personne n'est pas dans un état de détresse cognitive. La détresse cognitive correspond à un épisode où une personne perd de manière temporaire une partie de ses capacités cognitives, ce qui peut induire l'apparition de comportement anormal souvent lié à des troubles de la mémoire ou de l'orientation. La détection de ces épisodes permet aux systèmes assistants une intervention ciblée au moment de ces épisodes. Cette assistance peut prendre plusieurs formes, la plus simple étant l'établissement d'une communication avec la personne afin de confirmer son état et de proposer une ou plusieurs actions permettant un retour à une situation sans détresse.

Une façon de détecter ces épisodes est de s'intéresser aux anomalies dans l'exécution des actions que la personne entreprend. Un individu en situation de détresse cognitive va généralement mal réaliser certaines actions, réaliser des actions peu probables au vu de la situation ou du contexte, ou à l'inverse ne pas réaliser une action attendue (Sodemann *et al.*, 2012) (Geib et Goldman, 2003). Ce problème peut être vu comme un problème de détection d'action au sein d'une séquence, et donc peut être abordé comme un problème de reconnaissance de plan (Roy *et al.*, 2009).

Le fait de résoudre ce problème en utilisant une approche issue de la reconnaissance de plan pose plusieurs contraintes parmi lesquelles les deux suivantes : (1) la solution doit pouvoir inférer l'objectif initial de l'agent observé, ce qui peut être rendu plus compliqué par le bruit d'observation et le fait que, comme il y a un risque d'actions imprévues, les observations ne sont pas fiables (2) la solution doit être rapide afin de pouvoir aider une personne en détresse cognitive avant que la situation ne deviennent trop problématique.

Dans les chapitres 1 et 2, nous avons présenté PARC et surtout l'approche de reconnaissance de plan utilisant des filtres à particules. Cette approche est prometteuse pour une utilisation comme détecteur d'anomalies dans une séquence d'actions, et ce, pour trois raisons : (1) elle est rapide, (2) elle offre une estimation des probabilités des différents objectifs et des actions futures, ce qui peut être utilisé pour faciliter la détection, (3) elle supporte dans une certaine mesure les bruits d'observations.

Dans ce chapitre, nous proposons deux méthodes utilisant la reconnaissance de plan permettant de détecter des anomalies dans l'exécution de plans. La première méthode consiste à décrire le problème comme un problème bayésien dynamique et d'utiliser un estimateur de transition entre un état normal et un état d'anomalie comme moyen de détection. Celle-ci est utilisable avec n'importe quel système de reconnaissance d'objectif, tant que celui-ci fournit une prédiction des probabilités de chacun des objectifs et peut déterminer si une action fait partie des actions nécessaires pour atteindre un objectif ou non. La seconde méthode est plus spécifique au filtre à particules présenté au chapitre 1, mais elle peut être utilisée avec toute autre approche offrant une prédiction sur les actions futures. Elle se base sur une comparaison de cette prédiction à la nouvelle observation.

Nous avons évalué nos méthodes sur des domaines simulés et ceux-ci montrent la

pertinence de l'approche et son potentiel pour des applications concrètes. Cependant, la mise en pratique est rendue difficile par une certaine sensibilité au bruit qui rend limitée une intégration de ces approches dans PARC pour une utilisation réelle sur un robot mobile. Celles-ci resteraient pourtant pertinentes pour une utilisation dans un contexte moins bruité comme un environnement intelligent.

Ce chapitre est en partie basé sur notre article présenté au *workshop Robots for Health and Elderly Care* de l'*International Conference on Intelligent Robots and Systems* IROS-RoboHEC (Massardi et Beaudry, 2020) et est en cours de publication.

Ce chapitre se compose de la manière suivante. Dans la section 3.1, nous présenterons l'état de l'art en nous intéressant à trois problèmes similaires, la détection d'anomalies dans des actions (*Anomaly detection*), la reconnaissance d'abandon d'objectifs (*Goal abandonment*) et la reconnaissance d'engagement dans un plan (*Plan commitment*) Dans la section 3.2, nous présenterons notre première contribution, une approche générale utilisant la reconnaissance d'objectifs. En section 3.3, nous présenterons notre seconde contribution, une approche utilisant les prédictions d'observations. Ensuite en section 3.4, nous présenterons les résultats expérimentaux. Enfin en section 3.5, nous ferons une brève discussion de ces approches avant de conclure.

3.1 État de l'art en détection d'anomalies

La détection d'anomalies dans des actions, souvent appelée détection d'anomalies (*Anomaly detection*), peut être décrite de la manière suivante : sachant une séquence d'observations $O = \{o_1, o_2, ..., o_{t-1}\}$ et une observation o_t, peut-on déterminer que cette dernière est normale, c'est-à-dire attendue, ou non. On peut

généralement aussi exprimer le problème par l'équation suivante :

$$P(o_t|O) < \sigma \tag{3.1}$$

, où σ est un seuil qui correspond à la limite où une observation est considérée comme anormale. Ce seuil σ dépend du problème et de la situation.

Il existe de nombreux travaux en détections d'anomalies (Sodemann *et al.*, 2012) avec des applications dans des domaines divers comme la détection d'intrusion, la conception de systèmes de sécurité ou, ce qui est plus pertinent dans le cadre de cette thèse, la détection d'anomalies dans les ADLs (Yahaya *et al.*, 2019).

La plupart de ces travaux abordent le problème de détection des anomalies en utilisant des approches issues de la reconnaissance d'activités (Zhu *et al.*, 2012) (Sultani *et al.*, 2018) (Revathi et Kumar, 2017) (Yahaya *et al.*, 2019).

Une des techniques les plus efficaces pour ce problème est l'utilisation de l'apprentissage profond sur des séquences d'actions (Revathi et Kumar, 2017). Dans leurs travaux, les auteurs utilisent une description temporelle des actions. Cette description permet de caractériser une action avec deux éléments clés : (1) la durée de l'action, ce qui permet de détecter certain types d'anomalies, si une action est trop longue ou au contraire trop courte, (2) la position de l'action dans le temps par rapport aux autres actions, ce qui permet une analyse de la séquence des actions. Comme l'information de la position de l'action dans une séquence semble être une caractéristique déterminante dans la détection des anomalies, il semble raisonnable d'adresser ce problème de détection d'anomalies en utilisant une technique exploitant les séquences d'actions. Par conséquent, des approches issues de la reconnaissance de plan pourraient apporter des éléments de solutions.

Un des premiers travaux proposant une solution à un problème analogue à la détection d'anomalies en utilisant la reconnaissance de plan a été réalisé par Geib

et Goldman (Geib et Goldman, 2003). Dans cet article, les auteurs s'intéressent au problème de reconnaissance d'abandon d'objectifs (*Goal abandonment*). Ce problème consiste à reconnaître si un agent observé poursuit toujours le même objectif, action après action, ou si son objectif a changé et qu'il a abandonné un objectif précédent. Ce problème peut être vu comme un cas particulier de détection d'anomalies en ce sens que l'abandon de plan peut consister en une forme d'anomalies où une fois que l'anomalie a eu lieu, l'agent observé a immédiatement choisi un autre objectif à atteindre, étant donné la présence de nouvelles actions. Il est important de noter que cette approche est basée sur des bibliothèques de plans utilisant le même formalisme que ceux utilisés par PARC et par les filtres à particules présentés aux chapitres 1 et 2, sans les modifications apportées que sont le modèle de décision et la fonction de bruit.

L'approche proposée par Geib et Goldman fonctionne de la manière suivante : pour chaque objectif en cours d'un agent observé on calcule la probabilité que la nouvelle action observée fasse partie d'un plan permettant d'atteindre cet objectif. Si cette probabilité est sous un seuil prédéfini, on considère que l'objectif associé a été abandonné. Cette approche demande plusieurs hypothèses en plus de celle posée par l'utilisation d'une bibliothèque de plan : (1) On suppose l'équiprobabilité des actions au sein des actions possibles ; on suppose une observabilité parfaite, (2) on suppose une connaissance à priori du ou des objectifs de l'agent observé. Dans leur article, les auteurs ne précisent pas l'impact qu'aurait le fait de relâcher la dernière hypothèse en prenant comme entrée une probabilité des objectifs qui pourrait être obtenue par un système de reconnaissance d'objectifs.

Une autre approche proposant une solution à un sous-problème de la détection d'anomalies en utilisant la reconnaissance de plan est la détection d'engagement dans un plan (*Plan commitment*) (Pereira *et al.*, 2020b). Cette approche se base sur la reconnaissance de plan par planification (Ramırez et Geffner, 2009) et sur

l'utilisation de repères (*landmark*) (Pereira *et al.*, 2020a). Elle consiste à mesurer l'optimalité d'une action en fonction des objectifs en cours d'un agent observé. Si l'action est considérée suboptimale, on considère que l'agent observé a dévié de son objectif et ne suit plus l'objectif en cours. Cette approche peut être vue comme un problème d'abandon de plan et par conséquent de détection d'anomalies.

Une autre approche de la reconnaissance de plan qui n'est pas formellement de la détection d'anomalies, mais qui reste pertinente pour cet état de l'art vient des travaux de Bouchard et al. suivi de Roy et al. (Bouchard *et al.*, 2007) (Roy *et al.*, 2009). Dans leurs travaux, les auteurs utilisent la reconnaissance de plan pour inférer l'objectif de personnes âgées atteintes d'Alzheimer dans un environnement intelligent. Pour ce faire, les auteurs utilisent une approche basée sur des bibliothèques de plans où tous les plans, valides, c'est-à-dire liés à un objectif, et invalides, c'est-à-dire non liés à un objectif, sont explicités. En utilisant cette approche, un système est capable de déterminer l'objectif d'un individu ou si celui-ci est en état de détresse cognitive.

3.2 Approche par reconnaissance d'objectifs

On définit la détection d'anomalies dans l'exécution d'un plan par la réalisation d'une action improbable au vu du ou des objectifs que cherche à atteindre un agent observé. On ne fait pas l'hypothèse ici que le ou les objectifs sont connus à priori, ceux-ci pouvant être inférés à partir de la séquence d'action. Formellement parlant, si l'on a une séquence d'action $A = a_{t-1}, a_{t-2}...a_1$, on considère qu'il y a anomalie si $P(a_t|A) < \sigma$ où σ est un seuil prédéfini.

La différence entre cette définition et la définition de la détection d'anomalies présentées en section 3.1 vient du fait qu'ici on s'intéresse aux actions directement plutôt qu'aux observations de manière générale. Il existe un lien entre les actions

et les observations. On peut retrouver la probabilité d'une action à partir d'une observation si il existe une matrice de confusion ou une fonction de bruit tel que décrit en chapitre 1. Le problème décrit ici peut être vu comme à la fois un sous-problème de la détection d'anomalies et comme une généralisation de la détection d'abandon d'objectifs. Il s'agit d'un sous-problème de la détection d'anomalies, car ici on ne se concentre pas sur les actions en elle-même, notamment de savoir si une action a été exécutée correctement ou non. C'est une généralisation de la détection d'abandon d'objectifs, car ici on ne fait pas l'hypothèse qu'une action anormale vise à la réalisation d'un autre objectif. Celle-ci peut avoir été exécutée pour d'autres raisons : il pourrait s'agir d'une répétition d'une action précédente, d'un oubli ou de l'exécution d'une action en plus sans lien avec les objectifs en cours.

Afin de créer un modèle de représentation de l'état de normalité d'une personne observée, on introduit la variable B_t. Celle-ci est vraie si une personne est dans un état anormal à l'instant t, c'est-à-dire que $P(a_t|A) < \sigma$ comme dans la définition de détection d'anomalie présentée précédemment.

En appliquant un développement bayésien à cette variable, on obtient la formule suivante.

$$P(B_t) = P(B_{t-1})P(B_t|B_{t-1}) + P(\neg B_{t-1})P(B_t|\neg B_{t-1}) \tag{3.2}$$

La finalité de l'utilisation de ce modèle basé sur B_t est de se passer de l'expression de $P(a_t|X)$ où X est un état antérieur. On cherche à enlever ce terme, car il est long à calculer et que la plupart des solveurs de reconnaissance de plans, par bibliothèque de plan ou par planification, ne le calculent pas. Ce faisant on cherche à proposer une approche ne dépendant pas du solveur de reconnaissance de plan.

Afin de pouvoir utiliser ce modèle, nous proposons l'utilisation de l'hypothèse suivante : il existe une intention derrière une séquence d'actions, même si celle-ci contient, voir commence par, des anomalies. Dans notre représentation, cela consiste à considérer qu'à $t = 0$, donc avant la première action, le comportement est normal. Mathématiquement, cela peut s'écrire comme :

$$P(B_0) = 0 \tag{3.3}$$

Cette hypothèse est raisonnable et peu contraignante à la fois dans le cas général et dans le cas de détection de détresse cognitive. En effet, celle-ci n'a pas vraiment d'impact sur la détection même dans le cas où la première action est anormale.

Dans l'équation (3.2), les termes $P(B_{t-1})$ et $P(\neg B_{t-1}) = 1 - P(B_{t-1})$ peuvent être obtenus de manière itérative en utilisant l'hypothèse en (3.3). Reste à identifier les termes $P(B_t|B_{t-1})$ et $P(B_t|\neg B_{t-1})$ qui correspondent respectivement à transition d'un état d'anomalies à $t-1$ à un état d'anomalies à t, et la transition d'un état normal à $t-1$ à un état d'anomalies à t.

Si l'on considère la normalité comme l'exécution d'une séquence d'actions prédéfinie, alors à partir du moment où il y a une déviation dans cette séquence, toute nouvelle action devient nécessairement anormale. Autrement dit, si une action entraîne une situation d'anormalité, il n'y a pas de retour à la normale possible. Mathématiquement, cela s'exprime de la manière suivante.

$$P(B_t|B_{t-1}) = 1 \tag{3.4}$$

Ce principe de non-retour à la normale peut paraître contraignant pour une application en situation réelle. On pourrait donner comme exemple quelqu'un qui est en train de se faire une tasse de thé, mais au milieu du plan, prend le temps de se

moucher. En fonction de comment est modélisé la bibliothèque de plan ou le domaine de planification utilisé en reconnaissance de plan, ce cas pourra être traité comme une situation normale ou une anomalie. Dans le cas des bibliothèques de plan, si le fait de se moucher est un objectif à part entière, ou s'il a été modélisé comme une action excédentaire dans un modèle de bruit, alors la situation pourrait être reconnue comme normale. Dans le cas contraire, il s'agirait d'une anomalie. La contrainte que pose le principe de non-retour pèse donc sur le modèle.

Cette nouvelle équation nous permet d'utiliser la simplification suivante :

$$P(B_t) = P(B_{t-1}) + (1 - P(B_{t-1}))P(B_t|\neg B_{t-1}) \tag{3.5}$$

Le dernier terme à déterminer est donc $P(B_t|\neg B_{t-1})$, soit la probabilité de transition d'un état normal à un état anormal.

Dans cette première approche, nous proposons une généralisation de l'approche de Geib et Goldman. Pour une hypothèse e, c'est-à-dire un plan ou un plan partiel, une anomalie à t correspond à l'exécution d'une action à l'instant t qui ne fait pas partie des actions prévues par l'hypothèse, ou à la présence d'anomalies plus tôt dans la séquence. Cela peut s'écrire de la manière suivante :

$$B_t \wedge e = (a_t \wedge e \wedge (a_t \notin e)) \vee B_{t-1} \tag{3.6}$$

, où $(a_t \notin e)$ exprime le fait que a_t n'est pas dans les actions prévues par e

En utilisant cette équation, on peut poser l'équation de probabilité suivante :

$$P(B_t \wedge e \wedge a_t|\neg B_{t-1}) = P(a_t \wedge e \wedge (a_t \notin e)) \tag{3.7}$$

On généralise pour toutes les hypothèses e valables à l'instant t et pour toutes les actions a possibles à ce même instant t.

$$P(B_t|\neg B_{t-1}) = \sum_{a \in A} \sum_{e \in E} P(a \wedge e \wedge (a \notin e)) \tag{3.8}$$

Cette généralisation pour les actions est nécessaire dans le cas d'observations imparfaites où a n'est pas connu avec certitude.

$(a \notin e)$ n'étant pas une probabilité, mais un opérateur logique (une action fait ou ne fait pas partie d'un plan), on peut la retirer du terme $P(a \wedge e \wedge (a \notin e))$ et la remplacer par un filtre de Dirac δ

$$P(B_t|\neg B_{t-1}) = \sum_{a \in A} \sum_{e \in E} \delta_{a \notin e} P(a \wedge e) \tag{3.9}$$

Le calcul de $P(a \wedge e)$ peut être coûteux et la plupart des systèmes de reconnaissance de plan et d'objectif ne donnent au mieux que $P(e)$. On va par conséquent à nouveau utiliser Bayes :

$$P(a \wedge e) = P(a|e)P(e) \tag{3.10}$$

$P(a|e)$ peut représenter deux réalités : (1) il s'agit de la probabilité de a sachant une hypothèse dans le cas où le comportement est normal, cette probabilité ne nous intéresse pas ici vu l'utilisation du filtre de Dirac (quelle que soit sa valeur, in fine son poids dans l'équation est 0), (2) il s'agit de la probabilité de réaliser une action en cas d'anormalité sachant e.

Cette probabilité de réaliser une action en cas d'anormalité sachant e peut être obtenue de plusieurs manières, entre autres en ajoutant sa définition au modèle,

par apprentissage ou en utilisant une hypothèse d'indépendance entre les actions et les hypothèses de plans dans le cas de comportement anormal. Cette dernière hypothèse semble rationnelle au vu de notre définition de comportement anormal. Des hypothèses similaires sont utilisées par Geib et Goldman dans leurs travaux, avec des résultats pertinents (Geib et Goldman, 2003) (Geib et Goldman, 2009). Pour la suite, nous adopterons cette hypothèse d'indépendance entre les actions et les hypothèses de plans dans le cas de comportement anormal. Dans ce cas l'équation (3.9) prend la forme suivante :

$$P(B_t|\neg B_{t-1}) = \sum_{a \in A} \sum_{e \in E} \delta_{a \notin e} P(a) P(e) \tag{3.11}$$

En utilisant un raisonnement similaire à celui trouvé dans le chapitre 1 section 1.1, on peut généraliser des hypothèses e aux objectifs g.

$$P(B_t|\neg B_{t-1}) = \sum_{a \in A} \sum_{g \in G} \delta_{a \notin e} P(a) P(g) \tag{3.12}$$

Pour cette première approche, il reste 2 termes à déterminer : (1) le seuil de confiance σ à partir duquel le système détecte un comportement anormal, celui-ci dépend du problème et (2) la probabilité $P(g)$ pour chaque objectif $g \in G$. Pour $P(g)$ nous proposons d'utiliser comme estimateur la probabilité $P(g)$ à $t-1$ obtenue par le système de reconnaissance d'objectifs.

Cette approche ne dépend pas du solveur et ne demande pas d'informations en dehors du modèle, que ce soit les solveurs par bibliothèque de plan ou par planification inversée, à l'exception des probabilités $P(a|e)$ dans le cas de comportements anormaux et de $P(a)$ en fonction des observations. Ce problème peut être contourné par l'utilisation d'hypothèses courantes en reconnaissance de plan

comme considérer que toutes les actions sont équiprobables en cas d'absence d'hypothèses.

3.3 Approche par prédiction d'observations

L'approche présentée dans la section précédente est applicable aux systèmes de reconnaissance de plan, tant que ceux-ci produisent la probabilité des objectifs, des actions, ainsi que l'information si oui ou non une action est nécessaire pour réaliser un objectif. Cependant cette approche s'éloigne de la définition originale de la détection d'anomalies décrite dans l'équation (3.1).

Une seconde approche consisterait à utiliser directement cette équation. Cette seconde approche n'est possible que dans le cas où le système de reconnaissance de plan est capable de fournir les probabilités des observations futures en fonction des observations passées.

Le système de reconnaissance de plan utilisant des filtres à particules présenté dans le chapitre 1 fournit ces prédictions au travers du calcul des ENO. Nous proposons ici d'utiliser ces prédictions comme système de détection, en comparant la probabilité de l'ENO correspondant à l'observation en cours à un seuil prédéfini.

$$B_t = P(ENO = o_t) < \sigma \tag{3.13}$$

Il s'agit d'une application directe du système décrit en chapitre 1 au problème de détection d'anomalies. Le principe de calcul des ENO est décrit en section 1.4.

3.4 Évaluation

Comme pour le chapitre 2, il n'existe pas à notre connaissance de bibliothèque de plans publique permettant d'évaluer ces approches et de les comparer à d'autres approches. Une des raisons étant que les autres approches de détection d'anomalies se concentrent plus sur les actions que sur la séquence d'actions. Afin de montrer l'efficacité des 2 approches présentées en 3.2 et 3.3, nous avons construit des bibliothèques de plans similaires à celles introduites dans le chapitre 2.

3.4.1 Domaines

Nous avons créé deux ensembles de bibliothèques de plans tels que décrits aux chapitres 1 et 2. Pour rappel, celles-ci consistent en 5 objectifs, une profondeur de 4, des règles d'une longueur de 3 symboles, un nombre de règles possibles par symbole de 2 et un ordonnancement partiel à 33%. Pour le premier ensemble nous avons fixé le nombre d'actions de bas niveau à 100 et à 10 pour le second ensemble. Cette variation a pour objectif de mesurer l'impact de l'ambiguïté sur nos approches. En effet, une bibliothèque de plans avec plus d'actions sera moins ambiguë qu'une bibliothèque de plans avec moins d'actions, chaque action étant plus significative dans l'identification des plans et des objectifs.

Pour représenter des anomalies dans l'exécution d'un plan, on choisit une valeur t représentant l'instant où un agent observé arrête de suivre un plan. t doit être inférieure à la longueur totale du plan. A l'instant t, une action non valide est choisie aléatoirement. Toutes les actions réalisées après cet instant t sont choisies de manière aléatoire dans l'ensemble des actions possibles.

Nous avons réalisé deux ensembles d'expériences, le premier où tous les plans contiennent des anomalies et un second ensemble où les plans sont réalisés cor-

rectement jusqu'à la fin. Le premier ensemble servant à déterminer la capacité de détection d'anomalies de nos approches, le second la capacité à reconnaître qu'un plan se déroule de manière normale. Nous avons testé nos approches pour 4 valeurs de bruit, 0%, 10%, 20% et 30%. Pour les deux approches, nous avons utilisé le système de reconnaissance de plan utilisant des filtres à particules décrit au chapitre 1. Le nombre de particules fixé à 1000. Pour l'approche utilisant la prédiction des objectifs, nous avons fixé les seuils de reconnaissance de comportement anormaux à 1%, 3%, 5% et 10%. Au delà la précision de l'approche décroît rapidement.

Pour l'approche utilisant des prédictions d'observations, nous avons fixé les seuils à 1%, 3% et 5%. Nous avons limité ce test à 3 seuils car au delà de 3% la performance décroît rapidement. Pour chaque expérience, nous avons utilisé 100 instances et 10 plans par instance.

Nous avons mesuré les valeurs suivantes :

- La précision, c'est-à-dire le pourcentage de vrais positifs et de vrais négatifs
- Le taux de faux positifs. Dans le cas des expériences avec anomalies, un faux positif représente la détection d'une anomalie alors que celle-ci n'a pas encore eu lieu ;
- Le taux de faux négatifs. Celui-ci ne s'applique pas dans le cas des ensembles d'expériences ne contenant pas d'anomalies ;
- La latence moyenne, c'est-à-dire le temps moyen qui sépare l'apparition d'une anomalie de sa détection (exprimé en nombre d'actions). Celle-ci ne s'applique pas dans le cas des ensembles d'expériences ne contenant pas d'anomalies.

3.4.2 Résultats

Les tableaux 3.1 à 3.4 présentent les résultats pour la première approche, avec le tableau 3.1 les résultats pour un domaine avec 10 actions et des anomalies dans la séquence, le tableau 3.2 les résultats pour un domaine avec 100 actions et des anomalies dans la séquence, le tableau 3.3 les résultats pour un domaine avec 10 actions sans anomalies dans la séquence et le tableau 3.4 les résultats pour un domaine avec 100 actions sans anomalies dans la séquence. Les tableaux 3.5 à 3.8 présentent les résultats pour la seconde approche, dans le même ordre que précédemment.

Notre première approche utilisant des prédictions sur les objectifs donne de bons résultats en absence d'observations imparfaites avec une précision à plus de 90 % quelque soit le seuil choisi. En cas d'observations imparfaites, les performances diminuent rapidement. Dans le tableau 3.1 on peut voir que seuls les cas de bruit à 10% pour des seuils de 5 % et 10 % ont une précision supérieure à 50%. Au delà de 20% de bruit, aucun des seuils ne donne de bons résultats, avec la meilleure précision à environ 44 %.

Dans le tableau 3.1, la latence est supérieure à 1, ce qui signifie qu'il se passe en moyenne au moins une action entre le moment où une anomalie a lieu et le moment où elle est détectée. La seule exception est pour le seuil de 3 % à 20 % de bruit, mais ce résultat est probablement dû à la très faible précision de ce cas. On remarque que la plupart des cas d'erreur, c'est-à-dire lorsque la prédiction est erronée, sont des faux négatifs, (ie. le système ne détecte pas d'anomalie alors qu'il y en a) sauf dans le cas d'un seuil à 10% et 10% de bruits. Cela semble indiquer que notre première approche sous-estime les cas d'erreurs.

Le tableau 3.2 confirme les résultats du tableau 3.1 avec une précision élevée dans

Tableau 3.1 Détection d'anomalies en utilisant l'approche par prédiction d'objectifs pour un domaine avec 10 actions - cas avec des anomalies dans la séquence

	Bruit	Précision	Faux positifs	Faux négatifs	Latence
seuil = 1%	0%	0.914	0.011	0.075	1.492
	10%	0	0	1	-
	20%	0	0	1	-
	30%	0	0	1	-
seuil = 3%	0%	0.932	0.016	0.052	1.412
	10%	0.123	0.021	0.856	1.223
	20%	0.006	0.002	0.992	0.333
	30%	0	0	1	-
seuil = 5%	0%	0.935	0.016	0.049	1.517
	10%	0.553	0.1	0.347	1.72
	20%	0.087	0.02	0.893	1.425
	30%	0.002	0.003	0.995	2
seuil = 10%	0%	0.932	0.024	0.044	1.339
	10%	0.783	0.177	0.04	1.682
	20%	0.439	0.173	0.388	1.818
	30%	0.068	0.044	0.888	1.382

les cas sans bruits ou avec 10 % de bruit. On retrouve aussi une chute significative des performances pour un bruit à partir de 20 %. Dans le tableau 3.2 la latence est plus faible et la précision un peu plus grande, ce qui est cohérent avec un domaine contenant moins d'ambiguïtés car ayant plus d'actions. Dans le cas d'un seuil à 1%, la majorité des erreurs sont des faux négatifs. Pour les autres seuils il s'agit de faux positifs. Cela indique qu'avec moins d'ambiguïté, le système est plus précis en cas de bruit, mais détecte des anomalies là ou il n'y en pas.

Tableau 3.2 Détection d'anomalies en utilisant l'approche par prédiction d'objectif pour un domaine avec 100 actions - cas avec anomalies dans la séquence

	Bruit	Précision	Faux positifs	Faux négatifs	Latence
seuil = 1%	0%	0.999	0	0.001	0.1291
	10%	0.598	0.24	0.162	0.1605
	20%	0.166	0.139	0.695	0.1687
	30%	0.093	0.079	0.828	0.1505
seuil = 3%	0%	0.992	0.002	0.006	0.1159
	10%	0.692	0.307	0.001	0.1156
	20%	0.484	0.408	0.108	0.2128
	30%	0.256	0.441	0.303	0.2617
seuil = 5%	0%	0.994	0.004	0.002	0.1157
	10%	0.688	0.311	0.001	0.1424
	20%	0.515	0.485	0	0.1786
	30%	0.37	0.557	0.073	0.2351
seuil = 10%	0%	0.992	0.006	0.002	0.1381
	10%	0.696	0.304	0	0.1509
	20%	0.482	0.518	0	0.2116
	30%	0.4	0.6	0	0.21

Les tableaux 3.3 et 3.4 montrent que le système reste précis en absence de bruit, et ne détecte pas d'anomalies dans la séquence. La précision ne décroît pas strictement dans le tableau 3.3 et dans le tableau 3.4 pour un seuil de 1% en fonction du bruit. On observe une chute à 10 % de bruit avant une remonté. À partir de ces résultats, on peut déduire que la précision dépend au moins du seuil, du bruit et de l'ambiguïté de manière non linéaire. Les seuil optimaux étant différents entre les ensembles contenant que des anomalies et les ensembles sans anomalies,

Tableau 3.3 Détection d'anomalies en utilisant l'approche par prédiction d'objectifs pour un domaine avec 10 actions - cas sans anomalies dans la séquence

	Bruit	Précision	Faux positifs
seuil = 1%	0%	0.947	0.053
	10%	1	0
	20%	1	0
	30%	1	0
seuil = 3%	0%	0.95	0.052
	10%	0.879	0.121
	20%	0.989	0.011
	30%	1	0
seuil = 5%	0%	0.942	0.058
	10%	0.6	0.4
	20%	0.885	0.115
	30%	0.99	0.01
seuil = 10%	0%	0.928	0.072
	10%	0.429	0.571
	20%	0.448	0.552
	30%	0.847	0.153

la détermination d'un seuil pour un problème va dépendre de la concentration d'anomalies dans l'ensemble.

Le tableau 3.5 montre les résultats obtenus pour notre seconde approche, utilisant des prédictions d'observations, sur des domaines de 10 actions. La précision obtenue décroit quand la valeur du seuil augmente. Les meilleures valeurs de précision sont obtenues avec un seuil fixé à 1%. La précision est de manière générale moins bonne avec cette seconde approche, pour une latence équivalente. Cette seconde

Tableau 3.4 Détection d'anomalies en utilisant l'approche par prédiction d'objectif pour un domaine avec 100 actions - cas sans anomalies dans la séquence

	Bruit	Précision	Faux positifs
seuil = 1%	0%	0.993	0.007
	10%	0.448	0.552
	20%	0.674	0.326
	30%	0.813	0.187
seuil = 3%	0%	0.995	0.005
	10%	0.383	0.617
	20%	0.2	0.8
	30%	0.282	0.718
seuil = 5%	0%	0.992	0.008
	10%	0.364	0.636
	20%	0.131	0.869
	30%	0.11	0.89
seuil = 10%	0%	0.988	0.012
	10%	0.39	0.61
	20%	0.124	0.876
	30%	0.031	0.969

approche semble cependant obtenir de meilleurs résultats quand les observations sont bruitées et lorsque le seuil est de 1 %. À l'inverse de la première approche, les cas d'erreurs sont principalement de faux positifs.

Le tableau 3.6 montre les résultats pour des domaines de 100 actions. Toujours à l'inverse de la première approche, la précision diminue avec une ambiguïté plus basse. Les résultats confirment une certaine robustesse au bruit et confirment aussi que les cas d'erreurs sont principalement des faux positifs.

Tableau 3.5 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 10 actions - cas avec anomalies dans la séquence

	Bruit	Précision	Faux positifs	Faux négatifs	Latence
seuil = 1%	0%	0.929	0.037	0.034	1.41
	10%	0.802	0.175	0.023	1.546
	20%	0.657	0.323	0.02	1.715
	30%	0.612	0.369	0.019	1.794
seuil = 3%	0%	0.819	0.145	0.036	1.056
	10%	0.687	0.302	0.011	1.223
	20%	0.55	0.441	0.009	1.38
	30%	0.517	0.475	0.008	1.358
seuil = 5%	0%	0.67	0.31	0.02	0.9478
	10%	0.513	0.474	0.013	1.027
	20%	0.484	0.514	0.002	0.9463
	30%	0.403	0.594	0.003	1.127

Les tableaux 3.7 et 3.8 confirment les tendances générales : la précision de l'approche par prédiction d'observations est plus faible que la celle par prédiction d'objectifs et la majorité des erreurs sont des faux positifs.

Pour résumer les résultats, l'approche utilisant des prédictions sur les objectifs donne de meilleurs résultats sur la précision que l'approche par prédiction des observations. La première de ces deux approches à tendance à produire plus de faux négatifs, là où la seconde à tendance à produire plus de faux positifs.

Dans les deux cas, on obtient des résultats intéressants en cas d'absence de bruit ou de bruit faiblement prononcé, jusqu' à 10%. Cependant étant donné que sur PARC, le bruit d'observation à été empiriquement déterminé entre 20% et 35%

Tableau 3.6 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 100 actions - cas avec anomalies dans la séquence

	Bruit	Précision	Faux positifs	Faux négatifs	Latence
seuil = 1%	0%	0.846	0.15	0.004	0.1111
	10%	0.619	0.381	0	0.1179
	20%	0.452	0.548	0	0.1704
	30%	0.326	0.674	0	0.1933
seuil = 3%	0%	0.379	0.621	0	0.06069
	10%	0.26	0.74	0	0.05769
	20%	0.218	0.782	0	0.06881
	30%	0.185	0.815	0	0.07027
seuil = 5%	0%	0.185	0.815	0	0.03784
	10%	0.162	0.838	0	0.02469
	20%	0.14	0.86	0	0.03571
	30%	0.139	0.861	0	0.02158

dans le chapitre 2, une intégration complète de ces deux approches à PARC semble difficile. Il faut toutefois remarquer qu'à ce degré de précision, les deux approches semblent pertinentes pour des environnements avec moins de bruit, comme on pourrait en trouver dans des environnement intelligents, comme dans le cas des travaux de Bouchard et al. et de Roy et al. (Bouchard *et al.*, 2007) (Roy *et al.*, 2009).

3.5 Discussion

Les deux approches présentées dans ce chapitre nécessitent la définition à priori d'un seuil de confiance. Ce seuil dépend de plusieurs facteurs, notamment de la précision souhaitée, du bruit ou encore du nombre d'actions par rapport au nombre

Tableau 3.7 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 10 actions - cas sans anomalies dans la séquence

	Bruit	Précision	Faux positifs
seuil = 1%	0%	0.899	0.101
	10%	0.379	0.621
	20%	0.166	0.834
	30%	0.068	0.932
seuil = 3%	0%	0.698	0.302
	10%	0.311	0.689
	20%	0.116	0.884
	30%	0.06	0.94
seuil = 5%	0%	0.451	0.549
	10%	0.201	0.799
	20%	0.068	0.932
	30%	0.037	0.963

d'objectifs. La définition de ce seuil paraît être un problème non trivial.

Plusieurs techniques pourraient être utilisées pour adresser ce problème : (1) déterminer le seuil de manière empirique ou par apprentissage pour chaque nouveau problème, (2) trouver une méthode mathématique permettant de définir de seuil à partir d'une fonction à plusieurs paramètres, (3) trouver une variante des approches proposées ne nécessitant pas de seuil.

Nous nous sommes intéressés à ce dernier point. Nous proposons ici une ébauche d'approche basée sur l'utilisation des prédictions des objectifs, comme la première approche proposée, mais en n'utilisant pas de seuil de précision. L'idée derrière cette ébauche d'approche est de considérer la probabilité de transition

Tableau 3.8 Détection d'anomalies en utilisant l'approche par prédiction d'observation pour un domaine avec 100 actions - cas sans anomalies dans la séquence

	Bruit	Précision	Faux positifs
seuil = 1%	0%	0.846	0.154
	10%	0.329	0.671
	20%	0.107	0.893
	30%	0.032	0.968
seuil = 3%	0%	0.261	0.739
	10%	0.093	0.907
	20%	0.045	0.955
	30%	0.009	0.991
seuil = 5%	0%	0.063	0.937
	10%	0.028	0.972
	20%	0.011	0.989
	30%	0.005	0.995

$P(B_t|\neg B_{t-1})$ non plus comme la somme des probabilités de tous les cas d'anomalies possibles, mais comme la probabilité de l'anomalie la moins probable. Ceci peut s'interpréter de la manière suivante : on considère le cas comme normal tant qu'il existe une hypothèse valide permettant d'expliquer l'action observée, en absence d'hypothèse valide, la probabilité d'anomalie correspond à l'inverse de la probabilité de l'hypothèse la plus probable.

$$P(B_t|\neg B_{t-1}) = \min_{a \in A, g \in G} \{P(a)P(g) | a \notin g, P(a) \neq 0, P(g) \neq 0\} \tag{3.14}$$

En utilisant l'équation (3.14) appliquée à l'équation (3.9), on peut poser que si $P(B_t) > 0.5$ alors on est dans un comportement anormale. L'approche est inspirée

par la logique floue, même si ça n'en est pas formellement (Hájek, 2013). Ici comme en logique floue, cela revient à transformer l'opérateur de sommation en minimum.

Nous avons testé cette nouvelle approche sur les ensembles de test décrits dans la section 3.4. Ces résultats se trouve dans les tableaux 3.9 et 3.10.

Tableau 3.9 Détection d'anomalies en utilisant une approche floue - cas avec anomalies dans la séquence

	Bruit	Précision	Faux positifs	Faux négatifs	Latence
10 actions	0%	0.805	0.002	0.193	2.61
	10%	0.766	0.118	0.116	2.457
	20%	0.721	0.183	0.096	2.426
	30%	0.684	0.264	0.052	2.481
100 actions	0%	0.879	0.001	0.12	1.139
	10%	0.703	0.235	0.062	1.061
	20%	0.555	0.423	0.022	1.009
	30%	0.475	0.516	0.009	0.9453

Les résultats montrent que cette nouvelle approche est moins performante que les approches présentées en section 3.2 et 3.3. La précision est plus basse dans tous les cas, et la latence est plus importante. Cependant les résultats montrent que l'approche fonctionne, et qu'elle donne des résultats sans avoir à définir un seuil à priori.

3.6 Conclusion

Dans ce chapitre, nous proposons deux nouvelles approches à la détection d'anomalies utilisant des techniques issues de la reconnaissance de plan. La première consiste à utiliser un raisonnement bayésien couplé à un estimateur de transition

Tableau 3.10 Détection d'anomalies en utilisant une approche floue - cas sans anomalies dans la séquence

	Bruit	Précision	Faux positifs
10 actions	0%	0.955	0.045
	10%	0.504	0.496
	20%	0.253	0.747
	30%	0.159	0.841
100 actions	0%	0.996	0.004
	10%	0.452	0.548
	20%	0.193	0.807
	30%	0.064	0.936

d'un état normal à une anomalie. Le second utilise les prédictions d'observation de notre filtre à particules. Dans les deux cas, les probabilités obtenues sont comparées à un seuil prédéterminé.

La première approche peut être utilisée avec n'importe quel système de reconnaissance de plan tant que celui-ci fournit une prédiction des probabilités des objectifs de la personne observée et une liste des actions nécessaires pour réaliser un plan. La seconde approche utilise spécifiquement notre filtre à particules. Ces approches donnent des résultats intéressants et ne font pas appel à certaines hypothèses usuelles, comme la connaissance à priori d'un objectif particulier. De plus, l'utilisation de notre filtre à particules dans les deux approches permet d'obtenir des résultats en temps réel, les rendant particulièrement intéressantes pour des applications avec des personnes pouvant souffrir d'épisodes de détresse cognitive.

Bien que ces deux approches peuvent supporter quelques observations bruitées, les performances sont significativement impactées par un bruit supérieur à 10%. Dans le cas de PARC présenté au chapitre 2, le bruit obtenu de manière empirique

varie entre 20% et 35%, rendant ces approches de détection d'anomalies difficiles à mettre en pratique pour la détection d'ADLs avec notre système. Cependant, celles-ci pourraient trouver des applications dans le cas de systèmes moins bruités comme pour les environnements intelligents. Une autre option serait d'améliorer la qualité de la détection des activités de bas niveau en améliorant les performances de PARC.

Dans les travaux futurs, il serait intéressant pour le domaine de construire un ensemble de données correspondant à des ADLs dont certaines contiendraient des anomalies ou des épisodes de détresse cognitive. Ces ensembles offriraient de nouvelles perspectives dans la mise en application des approches décrites ici et permettraient de comparer différentes approches sur des situations réelles et non plus sur des simulations.

CHAPITRE IV

RECONNAISSANCE D'OBJECTIF ACTIVE

Comme vu dans les chapitres précédents, la reconnaissance de plan se fait dans le cas d'applications réelles, notamment sur des robots mobiles, dans un contexte où les observations sont imparfaites. La figure 4.1 illustre cette problématique. Il s'agit d'une vue subjective de PARC sur un robot mobile. En fonction de la distance à la scène observée, de l'orientation de la caméra et d'autres facteurs environnementaux, la qualité de l'observation et de la prédiction des intentions d'un agent observé peuvent varier grandement.

Une façon d'améliorer la précision de l'observation et de la prédiction est de chercher à repositionner les capteurs afin d'améliorer la prise de vue. Cette tâche est une tâche de planification automatique. Ce nouveau problème correspond au problème de reconnaissance d'objectif active (*Active Goal Recognition - AGR*). Elle se différencie de la reconnaissance d'objectif plus classique par le fait que l'observateur n'est plus passif, il n'attend pas d'observation, mais est actif dans ça démarche pour obtenir une prédiction de l'objectif de l'agent observé.

Bien que plusieurs auteurs aient déjà évoqué le fait de planifier des observations, l'AGR en tant que tel a été défini de manière indépendante à plusieurs reprises au cours des dernières années (Massardi et Beaudry, 2017b) (Massardi et Beaudry, 2018)(Amato et Baisero, 2019)(Shvo et McIlraith, 2020).

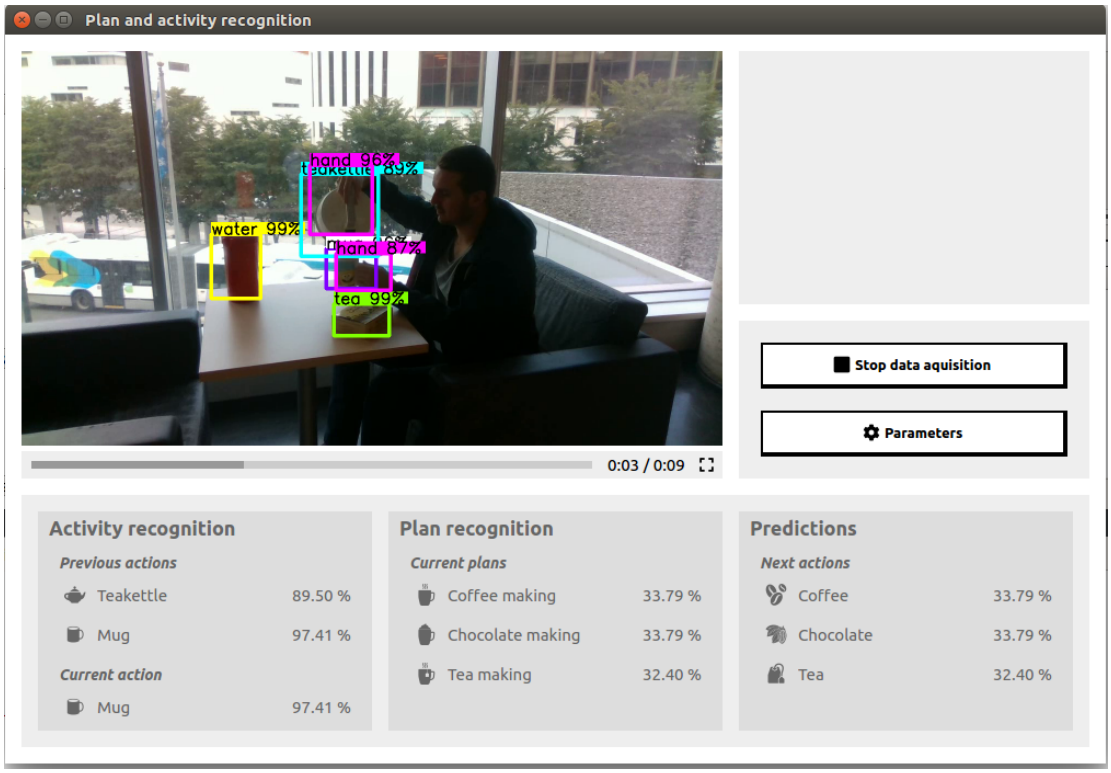


Figure 4.1 Vision subjective du robot mobile avec caméra embarquée

Il existe plusieurs applications possibles à la reconnaissance d'objectif active, principalement dans le cas de robots mobiles sans capteur extérieur, comme dans le cadre de cette thèse, et dans le cas de reconnaissance d'objectifs avec des ressources limitées. Dans le cas des robots mobiles, l'AGR peut offrir les avantages suivants :

— une amélioration de la précision, en améliorant le positionnement du robot on peut améliorer la prise de vue et éviter certains problèmes d'orientation ou d'occultation ;

— une limitation du nombre de mouvements, ce qui permettrait une amélioration de l'autonomie du robot

— une meilleur acceptation du robot par les personnes assistées, le robot pouvant avoir un comportement plus complexe et moins envahissant que sim-

plement suivre les personnes en permanence.

Un autre champ d'application de l'AGR est la détection de trajet (Ashbrook et Starner, 2002). Dans ce domaine d'étude, l'AGR permettrait de limiter le nombre d'observations depuis des capteurs mobiles, souvent des téléphones intelligents, ce qui permettrait une amélioration à la fois de l'utilisation de la batterie et de la gestion de la vie privée.

Pour résoudre les problèmes d'AGR, nous proposons ici d'utiliser une variante du IAMP *framework* (*Intention-Aware Motion Planning framework*) (Bandyopadhyay *et al.*, 2013). Ce framework s'appuie sur les processus de décision Markovien à observabilité mixte (*MOMDP pour Mixed Observable Markov Decision Processes*). L'utilisation des MOMDPs permet de représenter à la fois les variables d'état connues et inconnues, les actions possibles de l'observateur, ainsi qu'une représentation des intentions de l'agent observé, dans un seul modèle.

Nous avons testé notre solution sur des problèmes simulés et avons montré que cette solution est suffisamment rapide pour une utilisation quasi temps réel, en utilisant des solveurs hors lignes de MOMDPs à l'état de l'art. Elle permet de réaliser la planification des observations et la reconnaissance de plan au sein d'un seul système. Une des limites de notre solution vient de la définition des bibliothèques de plans : celle-ci n'est pas compatible avec la définition fournie aux chapitres 1, 2 et 3. Ce chapitre est principalement basé sur notre article accepté pour présentation au *workshop Plan, Activity, and Intent Recognition* de la conférence de l'*Association for the Advancement of Artificial Intelligence*(AAAI-PAIR) 2021.

Ce chapitre se compose de la manière suivante : en section 4.1 nous présenterons l'état de l'art ; en section 4.2 nous présenterons les MDPs, POMDPs et MOMDPs ; en section 4.3 nous présenterons le IAMP *framework* qui est à la base de notre approche ; en section 4.4 nous présenterons notre contribution principale, l'ex-

tension de ce *framework* que nous proposons pour résoudre des problèmes de reconnaissance d'objectif active ; ensuite en section 3.5 nous présenterons les résultats expérimentaux ; enfin en section 3.6 nous ferons une brève discussion de cette approche avant de conclure.

4.1 État de l'art en planification et reconnaissance d'objectif

Plusieurs travaux récents proposent d'utiliser la reconnaissance de plan ou d'objectif de manière conjointe avec de la planification afin de réaliser des tâches en commun (Levine et Williams, 2014) (Geib *et al.*, 2016). Par exemple, Freedman et al. proposent un framework pour une intégration directe entre la reconnaissance d'objectif et la planification basé sur des techniques de planification classique (Freedman et Zilberstein, 2017). Dans ces différents travaux, la reconnaissance de plan est utilisée comme une entrée d'information passive. L'observateur ne prend pas en considération les observations imparfaites et ne cherche pas à améliorer ses observations.

Une autre famille d'approches qui propose une combinaison entre la reconnaissance d'objectif et la planification est le GRD (*Goal Recognition Design*) (Keren *et al.*, 2014). Le GRD propose d'améliorer la reconnaissance d'objectif en modifiant l'environnement. Ces modifications de l'environnement ont pour but de faire réaliser des actions discriminantes le plus tôt possible dans l'exécution des plans. Ces actions permettant de reconnaître l'objectif plus facilement et plus tôt. Ces modifications ne sont possibles que si l'observateur a des moyens d'altérer l'environnement avant que l'agent observé ne débute son plan. Ce ne serait pas le cas pour un robot mobile à domicile par exemple.

Il existe plusieurs travaux utilisant des MDPs (*Markov Decision Process*) et des POMDPs (*Partially Observable Markov Decision Process*) pour faire de la plani-

fication en prenant en considération les intentions d'autres agents. Une partie de ces travaux se concentre sur les tâches coopératives où plusieurs agents partagent des objectifs en commun. C'est le cas pour les *Interactive-POMDP* (Gmytrasiewicz et Doshi, 2004), une généralisation des POMDPs. Dans ce *framework*, on utilise plusieurs POMDPs, un par agent, avec une seule fonction de récompense en commun. Ce modèle est considéré comme gourmand, chaque POMDP devant être résolu indépendamment à plusieurs reprises afin de faire converger les solutions. Fern et al. ont proposé un autre modèle, moins gourmand, dans le cas dans un environnement totalement observable en utilisant des MDP à objectifs cachés *(Hidden Goal MDP)* (Fern et Tadepalli, 2010). Une autre approche pour la planification avec d'autres agents dans l'environnement est l'IAMP (*Intention Aware Motion Planning*). Ce *framework* propose de résoudre ce problème de planification avec un seul modèle, un MOMDP, et ne suppose pas d'observabilité parfaite de l'environnement. Bien qu'intéressant pour résoudre les problèmes d'AGR, ce modèle n'est pas conçu pour réaliser de la reconnaissance d'objectif.

En ce qui concerne l'AGR en lui-même, il existe à notre connaissance 2 travaux dédiés à ce problème, l'utilisation de planification classique (Shvo et McIlraith, 2020) et l'utilisation de POMDPs (Amato et Baisero, 2019) . L'approche de Shvo et al. utilise une approche dérivée de la reconnaissance de plan par la planification. L'approche d'Amato et al. utilise un modèle très similaire à celui de l'IAMP et utilise les prédictions internes du modèle comme éléments de reconnaissance d'objectif, sans pour autant expliciter le *framework*.

4.2 Présentation des modèles MDPs

Dans cette section nous présentons les définitions des MDPs, POMDPs et MOMDPs. Il s'agit de modèles permettant de décrire des problème de planification non dé-

terministe.

4.2.1 MDPs

Un MDP est un modèle conçu pour permettre de représenter et résoudre des problèmes de planification avec des actions non déterministes dans un environnement totalement observable. Il s'agit d'un modèle à état fini discret avec des transitions probabilistes (Ghallab *et al.*, 2004).

Définition 4.1. *Un MDP est un tuple* (S, A, T, R) *avec :*

- S *un ensemble fini d'états ;*
- A *un ensemble fini d'actions ;*
- $T : S \times A \times S$ *une fonction de transition probabiliste de la forme* $T(s, a, s') = p(s'|s, a)$ *avec* $s, s' \in S$ *et* $a \in A$ *;*
- $R : A \times S \to \mathbb{R}$ *une fonction de récompense.*

La fonction de transition T représente la probabilité d'atteindre l'état s' depuis l'état s lorsque l'action a est réalisée. La fonction de récompense R correspond au gain numérique du modèle pour chaque paire d'actions/états (a, s). Avec un MDP, on suppose l'observabilité comme étant parfaite, c'est-à-dire que le système connaît l'état dans lequel il se trouve.

Étant donné la nature non déterministe des transitions, résoudre un MDP ne donne pas un plan à proprement parler, mais une politique. Le but est de trouver une politique maximisant la récompense totale du système. Une politique π se présente sous la forme $\pi : S \to A$ où pour chaque état $s \in S$ on associe une action $a \in A$.

La séquence d'états d'un MDP pouvant être infinie, une des méthodes permettant de maximiser la récompense du système est d'utiliser un escompte γ. Cet escompte

est d'une valeur comprise en 0 et 1 qui permet de faire converger la somme des récompenses. Plus l'escompte est proche de 1, plus la somme converge lentement mais sans trop pénaliser la récompense dans le futur, plus l'escompte est proche de 0 plus la somme converge vite, mais plus la récompense immédiate est favorisée.

La récompense espérée totale d'un MDP escompté est donnée par l'équation (4.1), avec r_t la récompense donnée obtenue à l'instant t par une politique π.

$$E(\sum_{t=0}^{\infty} \gamma^t r_t) \tag{4.1}$$

Pour une politique π donnée et un état s donné, la fonction de valeur $V^\pi(s)$ est la fonction qui donne le gain espéré si l'état de départ est s et qu'on applique indéfiniment la politique π. La politique optimale π^* est la politique qui maximise $V(s) = V^\pi(s)$ pour $\pi = \pi^*$ et pour tous les états $s \in S$. Cette politique la solution unique à l'équation de Bellman, donnée par l'équation (4.2).

$$V(s) = \max_{a \in A} \{R(a, s) + \sum_{s' \in S} T(s, s, a) V(s')\} \tag{4.2}$$

L'équation de Bellman peut être résolue de plusieurs manières. On trouve comme méthode de résolution l'itération de valeur (*Value Iteration*), qui consiste à partir d'une fonction de valeur donnée, itérer vers la fonction de valeur optimale, l'itération de politique (*Policy Iteration*), qui consiste à partir d'une politique donnée, itérer vers une politique optimale ou encore des approches comme PROST, qui utilise une approche d'exploration par simulation appelée UCT (*Upper Confidence Tree*) (Keller et Eyerich, 2012).

La figure 4.2 décrit un exemple de MDP à 3 états. Depuis S_1, le système peut réaliser 2 actions, a_1 et a_2. Si le système réalise a_1, alors il atteindra l'état S_2 avec une récompense de -10. S'il réalise l'action a_2, alors il pourra atteindre où S_2 avec une récompense de 1 où S_3 avec une récompense de -20. En connaissant les

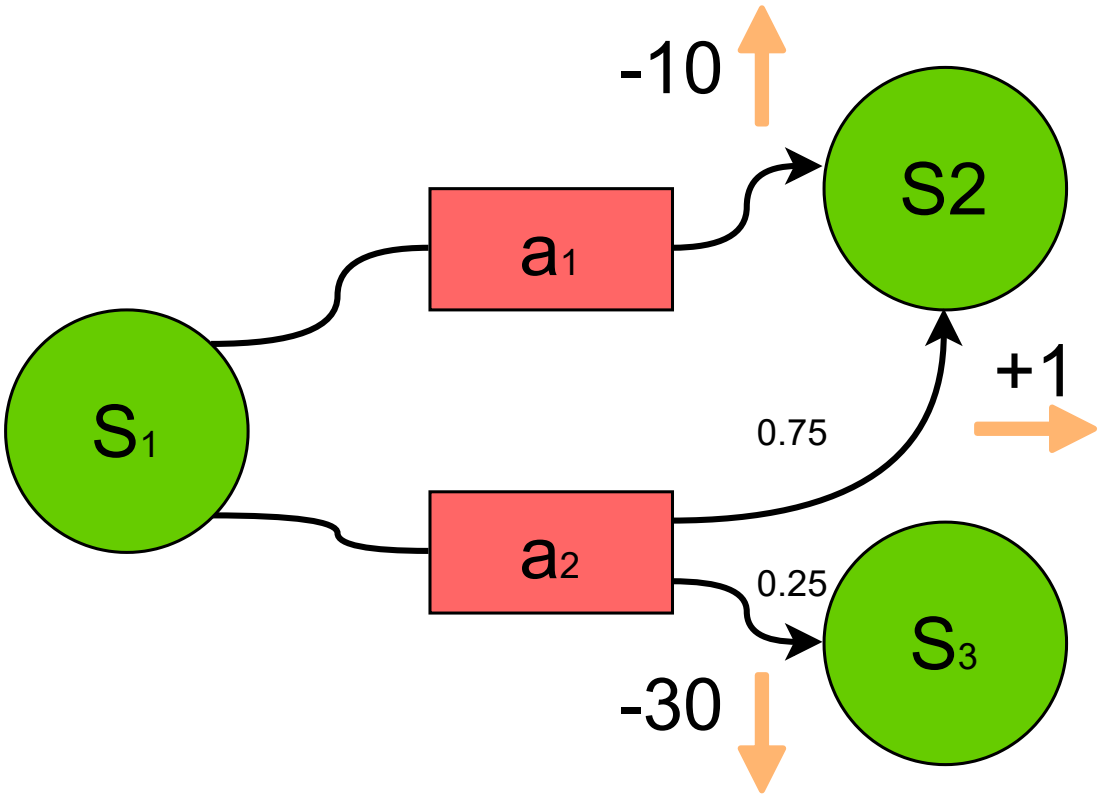


Figure 4.2 Exemple d'un MDP simple

probabilités de transition, la meilleure politique est réalisée l'action a_1 en S_1 pour une récompense espérée de $1 \times 0.75 + -30 \times 0.25 = -6.75$.

4.2.2 POMDPs

Un POMDP est une extension des MDPs où l'état courant du système est inconnu par le système lui-même (Ghallab *et al.*, 2004). Pour représenter cette incertitude, 3 nouveaux concepts sont introduits dans les POMDPs : (1) un ensemble d'observations O, (2) une fonction d'observation Z de la forme $Z : Z(o, s) = p(s|o)$ avec $s \in S$ et $o \in O$, (3) les états de croyances (*belief states*), un état de croyance étant une fonction de distribution sur l'ensemble des états du système.

Définition 4.2. *Un POMDP est un tuple* (S, A, T, R, O, Z) *avec :*

- S *un ensemble fini d'états ;*
- A *un ensemble fini d'actions ;*
- $T : S \times A \times S$ *une fonction de transition probabiliste de la forme* $T(s, a, s') = p(s'|s, a)$ *avec* $s, s' \in S$ *et* $a \in A$ *;*
- $R : A \times S \to \mathbb{R}$ *une fonction de récompense ;*
- O *un ensemble fini d'observations ;*
- $Z : Z(o, s) = p(s|o)$ *une fonction d'observation avec* $s \in S$ *et* $o \in O$.

Un POMDP peut être exprimé comme un MDP dont les états sont les états de croyance notés b. On peut alors réécrire l'équation de Bellman pour des POMDP en introduisant les variables b_a, l'état de croyance résultant de l'exécution de l'action a à l'état de croyance b et b_a^o, l'état de croyance obtenue après observation suivant b_a.

$$b_a(s) = \sum_{s' \in S} T(s', a, s) b(s') \tag{4.3}$$

$$b_a(o) = \sum_{s \in S} Z(o, s) b_a(s) \tag{4.4}$$

$$b_a^o(s) = Z(o, s) b_a(s) / b_a(o) \quad si \quad b_a(o) \neq 0 \tag{4.5}$$

Les équations (4.3), (4.4) et (4.5) donne le calcul de ces variables. Si on les applique à l'équation de Bellman en (4.2) on obtient l'équation suivante :

$$V(b) = \max_{a \in A} \{R(a, b) + \sum_{o \in O} b_a(o) V(b_a^o(s))\} \tag{4.6}$$

La résolution de l'équation (4.6) donne la politique unique optimale pour un

POMDP. Il existe plusieurs approches pour la résoudre. Parmi celles à l'état de l'art on retrouve SARSOP pour une résolution hors-ligne (Kurniawati *et al.*, 2008) et DESPOT pour une résolution en ligne (Somani *et al.*, 2013).

Un MOMDP est un POMDP donc les états sont représentés par plusieurs variables. Ces variables peuvent être représentées par deux sous-ensembles, un directement observable par le système, et un non observable par le système. Les MOMDP sont généralement plus simples à résoudre que les POMDPs, la gestion des états de croyance ne devant intervenir que pour le sous-ensemble des variables non observables (Kurniawati *et al.*, 2008).

Les POMDPs ont déjà été utilisés en reconnaissance de plan (Ramırez et Geffner, 2011). Ramírez et Geffner ont proposé une variante de leur approche de reconnaissance de plan en utilisant de la planification (Ramırez et Geffner, 2009) sur un agent raisonnant en utilisant des POMDPs. Le principe est de trouver les politiques optimales des POMDPs et de les comparer aux actions en cours. Le POMDP dont la politique optimale est la plus proche des actions observées est considéré comme le POMDP expliquant les actions de l'agent observé.

4.3 IAMP framework

Le IAMP*framework* est un *framework* utilisant des MOMDPS pour représenter des problèmes où il existe plusieurs agents dans l'environnement, avec au moins un agent contrôlé par le système et un agent non contrôlé par celui-ci. L'avantage de cette approche par rapport à d'autre (Gmytrasiewicz et Doshi, 2004) (Fern et Tadepalli, 2010), vient de l'utilisation d'un unique MOMDP pour la représentation et la résolution du problème.

D'un point de vue formel, un IAMP est un tuple $(X, Y, \Theta, A, O, T_X, T_Y, Z, R)$ avec :

— X l'ensemble d'états de l'agent principal (comprendre celui contrôlé par le système) ;
— Y l'ensemble d'états des autres agents ;
— Θ l'ensemble des intentions, ou des objectifs, des autres agents ;
— A l'ensemble des actions. Dans ce modèle, les actions n'impactent que X, l'agent contrôlé, Y ne dépend que de Θ et des transitions ;
— O l'ensemble des observations ;
— $T_X : X \times A \times X$ la fonction de transition de X de la forme $T_X(s, a, s') = p(s'|s, a)$ avec $s, s' \in X$ et $a \in A$;
— $T_Y : S \times \Theta \times S$ la fonction de transition de Y de la forme $T_Y(s', g, s) = p(s'|s, g)$ avec $s, s' \in Y$ et $g \in \Theta$;
— $Z : X \times Y \times O$ la fonction d'observations ;
— $R : A \times S \to \mathbb{R}$ la fonction de récompense.

Ce modèle peut être projeté en un POMDP (S, A, T, R, O, Z) avec $S = X \times Y \times \Theta$ et $T = T_X \times T_Y$, les autres termes étant transparents. Ce *framework* peut-être interprété comme une série séquentielle d'actions de la part de l'agent principal et de transitions de la part des autres agents. L'agent principal réalise une action $a \in A$. Le système atteint un état $s' \in S$. Le ou les autres agents entraînent une transition, dépendamment de $g \in \Theta$ et le système atteint un état final $s'' \in S$.

Dans ce *framework*, X et Y peuvent être indifféremment observables ou non observables. La variable d'état Θ est non observable. Elle peut être inférée en utilisant Y et T_Y.

4.4 Extension à la reconnaissance de plan

L'idée principale derrière notre approche est d'adapter le IAMP *framework* pour que celui-ci puisse expliciter les problèmes de reconnaissance d'objectifs. Par lui-

même, ce *framework* offre deux avantages principaux pour faire de la reconnaissance d'objectifs : (1) il y a déjà une représentation des objectifs cachés d'autres agents dans le modèle (2) il peut être résolu en une seule étape via sa projection en POMDP.

La limitation principale de ce framework pour résoudre des problèmes de reconnaissance de plan vient du fait que la croyance du système sur l'objectif ou le plan de l'agent observé est une valeur implicite. Par conséquent, il n'y a pas de fonction de récompense qui peut être associée à cette valeur. Comme aucune récompense n'est associée à cette croyance, le système ne cherche pas à lever d'ambiguïté sur le plan ou les objectifs de l'agent observé à moins que cette information ne soit nécessaire pour accomplir une autre tâche par le système.

Dans sa définition, le IAMP framework représente la suite des actions des autres agents par une fonction de transition liée à un objectif caché. Comme cette transition représente une collection de comportements, nous proposons de la considérer comme une bibliothèque de plans et par conséquent utiliser une approche de la reconnaissance d'objectifs basée sur des bibliothèques de plans.

4.4.1 Définitions

Dans notre représentation de l'AGR, nous utilisons la définition de bibliothèque de plans suivante.

Définition 4.3 *Une bibliothèque de plans PL est un tuple $PL = (A_{PL}, P, G, DM)$ avec :*

- *A_{PL} un ensemble d'actions ;*
- *P un ensemble de plans dans la forme $P \rightarrow a_0, a_1, ...a_n$ avec $a_i \in A_{PL}$;*
- *G un ensemble d'objectifs. Chaque objectif $g \in G$ est un sous-ensemble de*

P ;

— *DM un modèle de décision dans la forme* $DM(plan) = p(plan)$ *avec* $plan \in P$. *Le modèle de décision correspond à la probabilité à priori de chaque plan.*

Dans notre approche, nous définissons un plan comme une séquence d'action parfaitement ordonnée. Nous pourrions utiliser d'autres définitions, comme celles présentées au chapitre 1 de cette thèse. Cependant l'utilisation de telles définitions aurait un impact important sur les performances de la technique décrite ici, particulièrement dans le cas d'une utilisation avec une résolution hors-ligne.

Nous définissons un problème de reconnaissance d'objectif actif comme un problème de planification avec la définition suivante :

Définition 4.4 *Un domaine de planification d'un problème de reconnaissance d'objectif actif est un tuple* $AGR\text{-}PD = (S_{PD}, PL, A_{PD}, T, R, O)$ *avec :*

— S_{PD} *un ensemble d'états ;*

— PL *une bibliothèque de plans ;*

— A_{PD} *un ensemble d'actions ;*

— $T : S_{PD} \times A_{PD} \times S_{PD}$ *une fonction de transition de la forme* $T(s, a, s') = p(s'|s, a)$ *avec* $a \in A_{PD}$ *et* $s, s' \in S_{PD}$;

— $R : S_{PD} \times A_{PD} \to \mathbb{R}$ *une fonction de récompense ;*

— $Z : S_{PD} \times PL \times O$ *une fonction d'observation sur l'ensemble des actions* A_{PL}, *de la forme* $Z(a, s, a') = p(a'|a, s)$ *avec* $a, a' \in A_{PL}$ *et* $s \in S_{PD}$.

Nous insistons sur le fait que A_{PL} et A_{PD} sont deux ensembles d'actions distinctes. Le premier représente les actions de l'agent observé et le second l'ensemble des actions de l'observateur. R est utilisée pour favoriser ou pénaliser certains comportements. Par exemple elle peut servir à pénaliser l'utilisation inutile de capteurs, ceux-ci pouvant amener à l'utilisation inutile de ressources, comme l'utilisation de batteries. De manière générale, l'objectif de l'AGR est de trouver un objectif

caché $g \in G$ le plus rapidement possible tout en maximisant la récompense.

Definition 4.5 *Un problème de reconnaissance d'objectif actif est un tuple* $P = (AGR - PD, S_0)$ *avec :*

- $AGR - PD$*, un domaine de planification d'AGR ;*
- s_0*, un état initial* $s_0 \in S_{PD}$ *.*

4.4.2 MOMDP Framework pour l'AGR

Afin que le IAMP *framework* soit capable de supporter des problèmes de reconnaissance d'objectif, nous proposons de rendre explicite les croyances que l'observateur a des plans possibles de l'agent observé. Pour ce faire, nous proposons l'ajout de 4 nouveaux éléments dans le modèle :

- E_G, le plan supposé. Cette variable représente la croyance du système sur quel est le plan le plus probable. Comme E_G représente une croyance du système, il est totalement observable ;
- A_E, l'ensemble des actions permettant un changement sur E_G ;
- $T_E : E_G \times A_E \times E_G$, la fonction de transition sous la forme $T(s, a) = s'$ avec $s, s' \in E_G$ et $a \in A$. Comme E_G représente toujours une croyance, la transition est déterministe ;
- $R_E : E_G \times G \rightarrow R$, une fonction de récompense associée à E_G. Comme ici on cherche à faire de la reconnaissance d'objectif, la fonction de récompense devrait renvoyer une valeur positive si pour un objectif g, $e_G = g$ avec $e_G \in E_G$, et une valeur négative ou nulle dans le cas contraire.

En utilisant les définitions 4.3, 4.4 et le IAMP *framework* présenté plus tôt, on définit notre MOMDP *framework* pour l'AGR comme un tuple $(X, Y, G, E_G, A, A_E, T_X, T_Y, T_E, O, Z, R, R_E)$ avec :

- $X = S_{PD}$ de la définition 4.4. X peut être partiellement ou totalement

observable en fonction du problème ;

— $Y = A_{PL}$ de la définition 4.3. Si Y est totalement observable, alors on serait dans un cas d'observabilité parfaite des actions de l'agent observé et par conséquent l'utilisation de l'AGR serait inutile. Par conséquent, on suppose Y comme étant partiellement observable même si techniquement ce framework supporterait l'inverse ;

— $G = P$ de la définition 4.3. Cet ensemble représente les plans cachés de l'agent observé ;

— $E_G = G$ de la définition 4.3 ;

— $A = A_{PD}$ de la définition 4.4 ;

— A_E de la section précédente ;

— $T_X = T$ de la définition 4.4 ;

— T_Y la fonction de transition entre 2 actions de l'agent observé ;

— T_E de la section précédente ;

— $O = Y$ de la section précédente ;

— $Z = Z$ de la définition 4.4 ;

— $R = R$ de la définition 4.4 ;

— R_E de la section précédente.

Le modèle de décision DM est utilisé à l'état initial afin d'avoir une estimation des probabilités de chaque objectif $g \in G$.

Comme pour le IAMP *framework*, ce *framework* peut être projeté comme un POMDP (S, A, T, R, O, Z) avec $S = X \times Y \times G \times E_G$, $A = A \times A_E$, $T = T_X \times T_Y \times T_E$, $R = R \times R_E$, $O = O$ et $Z = Z$.

La figure 4.3 est une représentation schématique du processus de reconnaissance d'objectif active au cours d'une seule transition de l'état initial $s \in S = (X, Y, G, E_G)$ à un état final $s''' \in S = (X, Y, G, E_G)$. Comme pour le IAMP *framework*, cette transition peut être représentée grâce à des états intermédiaires. Le système tran-

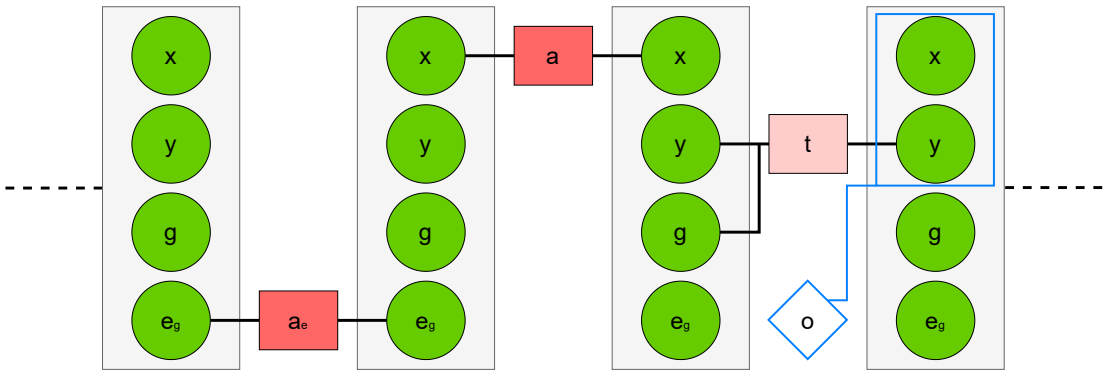


Figure 4.3 Représentation du processus de reconnaissance d'objectif active par MOMPD

site de l'état s à l'état s' par l'exécution d'une action a_e de l'ensemble A_E. Cette action n'impacte que le sous-état $e_g \in E_G$. Le système reçoit une récompense R_e en fonction de si e_g correspond à g. La seconde et la troisième transition intermédiaires concernent l'agent principal et l'agent observé de manières similaires au IAMP *framework*. Le système transite le l'état s' vers l'état s'' par une action réalisée par l'agent principal, suivie d'une transition de s'' à s''' en appliquant une transition de l'agent observé. L'observation o est émise à s''', avant un possible changement de l'estimation de l'objectif.

4.4.3 Exemple

La figure 4.4 représente un problème d'AGR. Dans cet exemple, un agent observé peut se rendre à trois emplacements différents : 7, 11 et 13. Le but ici est de reconnaître l'objectif de l'agent observé tout en minimisant le nombre des observations à faire. On suppose ici un observateur avec un capteur parfait qui peut être allumé ou éteint. L'agent observé ne peut faire qu'une action paramétrée, $goto_X$, qui lui permet de se déplacer à la case X si il se trouve sur une case adjacente. Pour simplifier l'écriture, cette action sera uniquement nommée par la

case de destination ($goto_2$ deviendra 2).

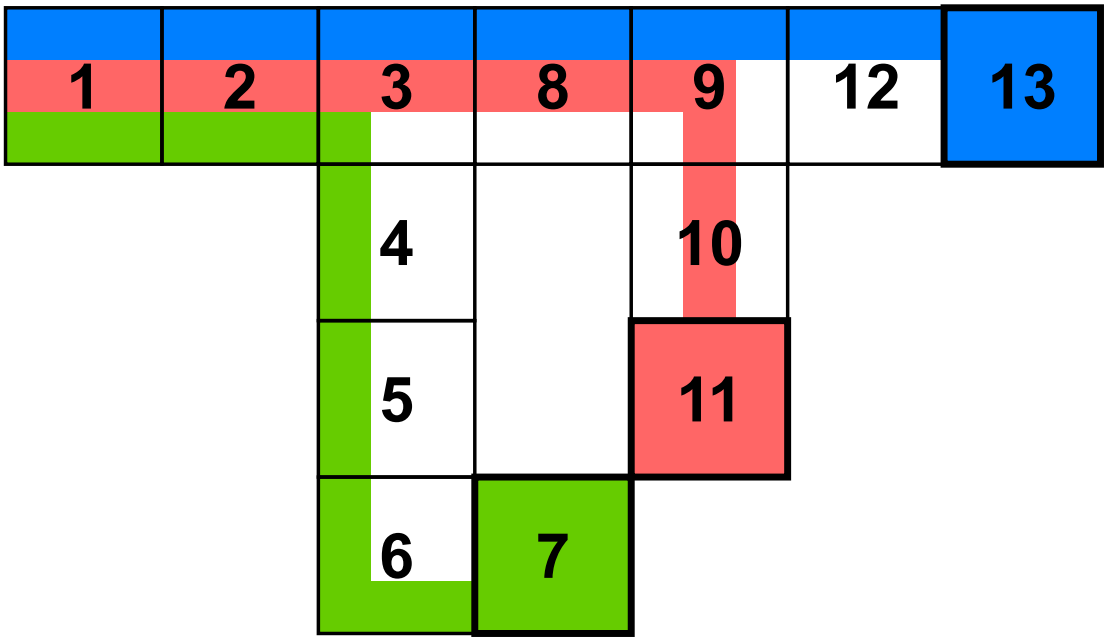


Figure 4.4 Exemple de problème d'AGR : reconnaissance d'objectifs en limitant le nombre d'observations

En utilisant la définition des problèmes d'AGR décrite précédemment, on obtient une bibliothèque de plans de 12 actions, 3 objectifs et 3 plans. Sans information supplémentaire sur le modèle de décision, on suppose une distribution uniforme des probabilités des plans.

Comme le but est de limiter le nombre d'observations, on représente notre observateur avec deux états : $SensOn$, qui représente l'état où le capteur est allumé et où l'observateur peut recevoir une observation et $SensOff$ dans le cas inverse. La transition de l'état $SensOn$ à $SensOff$ se fait par une action $switch$. L'observateur peut réaliser aussi l'action $wait$, pour laquelle il reste dans l'état en court.

L'ensemble des observations O correspond aux 13 états différents dans lequel peut

se trouver l'agent observé. On ajoute à cet ensemble l'état *noObs* qui correspond à l'absence d'observation dans le cas où le capteur est éteint, c'est-à-dire à l'état $SensOff$. Comme on suppose le capteur parfait, la fonction d'observation Z renvoie toujours l'état correspondant à l'exception des cas où le capteur est éteint et dans ce cas émet l'observation *noObs*. En ce qui concerne la fonction de récompense, nous proposons de pénaliser les observations par une valeur de -1 si l'observateur est dans l'état *sensOn*. L'état initial de l'observateur est à $SensOff$.

Nous devons maintenant définir les ensembles de valeurs de 4 variables nécessaires à la reconnaissance d'objectifs : E_G, A_E, T_E et R_E. E_G correspond aux objectifs possibles, c'est-à-dire 7, 11 et 13. A_E correspond aux transitions de E_G, c'est-à-dire $\{cheg_7, cheg_11, cheg_13\}$. Comme le but ici est de déterminer l'objectif de l'agent observé tout en minimisant le nombre d'observations, nous proposons d'utiliser une récompense un ordre de grandeur plus fort que la pénalité donnée en cas d'observation. Grâce à cela, le système devrait favoriser la précision de la reconnaissance d'objectif. Nous proposons une valeur $R_E = +10$ si $e_g = g$ avec $e_g \in E_G$ et $g \in G$. De manière arbitraire, nous fixons l'état initial de e_g à 11.

En utilisant la représentation par MOMDP des AGRs présenté à la section 4.4, on peut résoudre ce problème en utilisant n'importe quel solveur de MOMDP. Dans cet exemple, la politique optimale obtenue est la suivante : *wait* jusqu'à $y = 3$, puis *switch*. Si $o = 4$, $cheg_7$, *switch* et *wait* jusqu'à la fin, si $cheg_11$, *switch*, puis *wait* jusqu'à $y = 8$, puis *switch*. Si $o = 12$, $cheg_13$, *switch* puis *wait* jusqu'à la fin, sinon *switch* puis *wait* jusqu'à la fin. Cette politique a une espérance de récompense de 44.66 obtenue en utilisant SARSOP. En utilisant cette politique, la précision et la vitesse de la reconnaissance d'objectif sont les mêmes que dans le cas ou toutes les observations auraient été faites, mais elle est réalisée ici avec seulement 2 observations au lieu de 7.

4.5 Évaluation

Nous avons testé notre approche sur 2 ensembles de domaines. Le premier ensemble simule un agent se déplaçant sur une grille avec plusieurs destinations possibles, le but ici est de déterminer la destination tout en limitant le nombre d'observations. Le second problème simule un robot mobile devant réaliser de la reconnaissance d'objectif. Les expérimentations ont été réalisées avec le solveur de MOMDP SARSOP (Kurniawati *et al.*, 2008).

4.5.1 Problème 1 : Détection de trajets avec limitation de ressources

Dans ce problème, l'agent observé se trouve sur une grille avec un observateur ayant à sa disposition un capteur bruité. L'observateur doit déterminer la destination de l'agent observé, le tout en minimisant le nombre d'observations. Ce problème est assez similaire à celui présenté dans la section précédente (figure 4.4). La différence principale étant que le capteur ici est bruité.

La grille a une profondeur de 4 et une largeur de 10 avec un total de 40 états dans l'ensemble Y. Tous les plans sont d'une longueur de 10 actions, représentée ici par une succession de cases. Les plans sont générés de manière aléatoire. Nous interdisons le retour à une case déjà visité. Nous avons réalisé les tests en variant le nombre de plans de 3 à 6 pour 3 niveaux de bruits différents à 0%, 25% et 50% d'observations erronées.

Le tableau 4.1 présente un résumé des résultats obtenus. Dans ce tableau nous définissons la convergence comme le nombre moyen d'actions à réaliser par l'agent observé avant que l'observateur infère le bon plan. Dans ce tableau, nous présentons à la fois la convergence dans le cas actif, c'est-à-dire en utilisant le *framework* décrit dans ce chapitre, et la convergence passive, c'est-à-dire dans le cas où le

Tableau 4.1 Temps de calcul moyen, nombre d'observations moyennes, vitesse de convergence moyenne en mode actif et vitesse de convergence en cas d'observation complète pour la détection de trajets

	nb. plans	3	4	5	6
bruit 0%	CPU (s)	0.01	0.01	0.07	0.18
	avg. observation	1.67	2.25	2.8	3.4
	conv. active (%)	40	40	59	78
	conv. passive (%)	40	40	58	76
bruit 25%	CPU (s)	0.04	0.04	0.844	1.49
	avg. observation	1.6	1.8	2.1	2.2
	conv. active (%)	27	45	57	63
	conv. passive (%)	27	43	56	62
bruit 50%	CPU (s)	0.09	0.09	3.0	4.3
	avg. observation	1.6	1.7	1.9	2.0
	conv. active (%)	27	38	48	59
	conv. passive (%)	27	37	47	58

capteur est allumé en permanence. Le nombre d'observations dans le second cas est égal au nombre d'actions réalisées, c'est-à-dire 10.

Dans le cas de l'approche décrite dans ce chapitre, le nombre d'observations est inférieur à 10 dans tous les cas. Le système se comporte de la manière attendue, si une observation ne fournit pas d'informations ou si celle-ci n'est pas suffisamment discriminante, le système évitera de réaliser cette observation. Dans cette expérimentation, ce nombre d'observations ne dépasse pas une moyenne de 3.4 pour le cas contenant 6 plans différents pour 0% de bruit. De manière contre-intuitive, la convergence et le nombre d'observations diminuent avec l'augmentation du bruit. Ceci peut s'expliquer par la quantité d'information obtenue par chaque observation, ici le bruit peut fournir une information sur la position de l'agent observé, même en cas d'erreur d'observation.

La limitation du nombre d'observations a un impact limité sur la vitesse de convergence. La différence entre la convergence active et la convergence passive est d'au plus 2%. Cette marge peut encore être réduite en augmentant la valeur de récompense R_E dans le cas de détection correcte d'un objectif, mais cela entraînerait une augmentation du nombre d'observations.

Le temps de calcul moyen de la politique optimale varie de 0.01 seconde à 4.3 secondes en fonction du nombre de plans et du pourcentage de bruit. Comme attendu, le temps de calcul augmente de manière exponentielle en fonctions de ces deux paramètres.

4.5.2 Problème 2 : positionnement d'un robot mobile

Ce problème représente le cas d'un robot mobile dans un environnement fermé, comme un intérieur ou une pièce. Dans cet environnement, l'observateur est un robot pouvant se déplacer de position en position. Celui-ci n'a pas de capteur

extérieur, tous les capteurs sont embarqués. L'objectif pour cet observateur est de reconnaître le plan d'un agent observé le plus rapidement possible, tout en limitant son nombre de déplacements.

L'environnement est décrit par une grille de 5 par 5. Le robot peut se déplacer de case en case en utilisant l'action *forward*. Le robot peut aussi changer son orientation en utilisant les actions *orientationUp* et *orientationDown*. Ces actions changent l'orientation du robot d'une case, dans le sens antihoraire et horaire respectivement. Le robot est doté de capteurs bruités qui peuvent observer une action sur une profondeur de 3 cases en face du robot, avec une précision décroissante de 75%, 50% et 25%. Le robot peut observer les cases immédiatement à proximité des deuxième et troisième cases avec une précision de 20% et 25% respectivement.

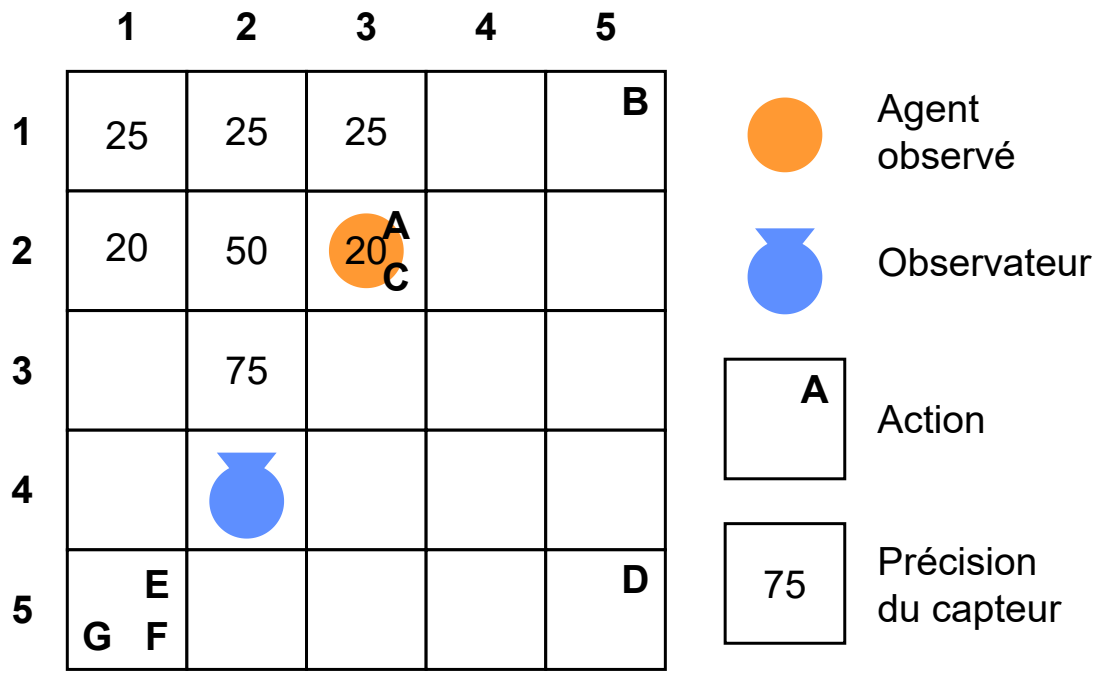


Figure 4.5 Représentation du problème d'AGR pour un robot mobile

Dans cette expérience, l'agent observé peut réaliser des plans de longueur de 6

Tableau 4.2 Temps de calcul, états de croyance et espérance de récompense pour le problème du robot mobile

	CPU (s)	États de croyance	E[Récompense]
Moyenne	21.608	10,400	30.448

actions. Les actions sont sélectionnées de manière aléatoire sur un ensemble de 7 actions possibles. Ces actions doivent être réalisées à des coordonnées spécifiques sur la grille. Ces coordonnées sont au nombre de 4. Ceci signifie que plusieurs actions peuvent être réalisées sur la même case. La figure 4.5 est une illustration du problème.

Nous avons décidé d'une récompense de $+10$ pour chaque fois que le robot reconnaît correctement l'objectif de l'agent observé, -2 si le robot se déplace d'une case et -1 dans le cas d'une rotation. Nous avons testé pour 100 domaines différents générés aléatoirement à partir des spécifications décrites ici.

Dans cette expérience, la politique optimale est calculée en environ 21 secondes pour 10 000 états de croyances. Cette rapidité de calcul rend cette approche utilisable dans des situations réelles. Cette politique est calculée hors-ligne et peut être réutilisé tant que l'environnement de change pas.

Dans la plupart des politiques obtenues, le robot se dirige vers la coordonnée de l'action suivante la plus probable afin d'y attendre l'agent observé et de pouvoir observer une action. Dans la majorité des tests, le robot réalisera cette opération de 2 à 3 reprises avant de considérer qu'il est suffisamment confiant concernant l'objectif caché de l'agent observé. Par la suite, le robot attendra jusqu'à la fin du test. Dans le cas où la prochaine action la plus probable est non discriminante, le robot ira aux coordonnées les plus proches de la prochaine action discriminante.

Dans certains cas et en fonction des coordonnées des actions, le robot peut se placer entre 2 coordonnées afin de minimiser le nombre de mouvements, même si cela augmente le risque d'une mauvaise observation pour au moins la première observation. Ce comportement change dans le cas ou la fonction de récompense R_e est augmentée. Dans ce cas le robot favorisera la précision au détriment de la limitation du nombre de mouvements.

4.6 Discussion

Ce *framework* donne de bons résultats pour le problème d'AGR pour plusieurs raisons : (1) bien qu'a la base conçue pour la planification de mouvements (*motion planning*), il s'agit d'un modèle discret qui s'adapte bien à la reconnaissance d'objectifs, (2) le modèle est synchrone et alterne entre des étapes de reconnaissance d'objectifs, de planification et d'observations, (3) le modèle a une profondeur finie, une fois que l'agent observé réalise la dernière action de son plan, le système a terminé.

Ce dernier aspect de notre approche peut causer des difficultés en fonction du solveur de MOMDP choisi, la plupart d'entre eux ne gérant pas des problèmes à horizon fini. Dans le cas de SARSOP, nous avons dû utiliser un état de garage vers lequel pointer tous les états dépassant la fin du problème afin de simuler un problème infini. Un autre problème rencontré concerne l'utilisation de la fonction de remise (*discount function*). Cette fonction est utilisée dans les MOMDPs afin de faciliter la résolution. Il s'agit d'une valeur inférieure à 1, qui est appliquée comme coefficient sur la récompense. Dans notre cas, l'utilisation de la remise signifie une variation entre la récompense théorique et la récompense obtenue. Toujours concernant les solveurs de MOMDPs, notre *framework* est facile à exprimer mais entraîne un nombre élevé d'états possibles, beaucoup n'étant pas atteignables.

Dans le cas de solveurs comme SARSOP, tous les états sont générés en mémoire à priori, ce qui peut entraîner un temps de chargement du modèle long, même si le temps de résolution du problème est rapide.

Ce *framework* pour l'AGR peut être étendu pour supporter le cas plusieurs agents observés. Dans ce cas, chaque agent observé i peut être représenté par un l'ensemble de variables suivant :

— un ensemble d'objectifs G_i ;

— une fonction de transition $T_{Y,i}$;

— une prévision d'objectif $E_{G,i}$.

En utilisant cette écriture, si l'on reprend le *framework* présenté à la section section 4.4.2, on obtient $G = \prod G_i$, $T_Y = \prod T_{Y,i}$ et $E_G = \prod E_{G,i}$. Bien que facile à exprimer, cette extension pour le cas multi-agent est gourmande à la fois en mémoire et en temps de calcul.

Ce *framework* a été utilisé comme tout premier solveur de reconnaissance de plan dans PARC (chapitre 2). Celui-ci a été pleinement intégré et présenté à plusieurs reprises (Massardi et Beaudry, 2017b) (Massardi et Beaudry, 2017a) (Massardi et Beaudry, 2018). Dans cette implémentation, le solveur génère une politique hors-ligne qui était réutilisée par PARC. Il existe deux versions, une passive, où l'observateur ne réalise aucune action à part ses prévisions sur les actions futures et l'objectif de l'agent observé, et une version active qui indique quelles actions mériteraient d'être observées pour améliorer les prédictions. Ce solveur a été abandonné au profit des filtres à particules, car ceux-ci sont plus rapides et ont une meilleure expressivité.

4.7 Conclusion

L'approche pour l'AGR présenté ici peut à la fois améliorer la reconnaissance de plan et limiter le nombre d'observations inutiles. Le *framework* de MOMPD que nous proposons ici est une extension du *framework* d'IAMP. Celui-ci donne des résultats intéressants dans et peut-être appliquer à plusieurs problèmes et supporte les d'observations imparfaites, prérequis à une application réelle.

La limitation principale de cette approche est que celle-ci demande une définition de plans comme une séquence totalement ordonnée d'actions. Dans des travaux futurs, il serait intéressant de supporter des modèles de plan plus expressif, comme les arbres de plans partiellement ordonnés (voir Chapitre 1). Une approche naïve de l'utilisation de ces arbres ne fonctionne pas en utilisant SARSOP comme solveur dû à une explosion combinatoire importante. Plusieurs solutions peuvent être explorées : (1) utiliser un solveur en ligne comme DESPOT plutôt que SARSOP, (2) utiliser de l'apprentissage par *Q-learnig* (Watkins, 1989) ou *Deep Q-learning* (Mnih *et al.*, 2015), (3) abandonner l'utilisation d'un modèle unique qu'exploite notre *framework* pour un modèle avec 2 composantes : planificateur rapide et un solveur de reconnaissance de plan lui aussi rapide.

CONCLUSION

Les travaux présentés dans cette thèse concernent la reconnaissance de plan, un sous-domaine de la planification en Intelligence Artificielle (IA). L'objectif de ces travaux était de proposer des approches de reconnaissance de plan capables d'être utilisées sur des robots mobiles. L'application de la reconnaissance de plan sur des robots mobiles permettrait aux robots d'apporter une assistance plus ciblée en ayant une meilleure compréhension des intentions de la personne assistée, ainsi qu'une évaluation des besoins de celles-ci, que le robot pourrait combler.

Ces travaux ont été motivés par les débouchés qu'apporteraient les robots à une problématique à laquelle se confrontent les pays développés, le vieillissement de la population. Dans ce contexte d'application, des robots assistants mobiles apporteraient un moyen d'améliorer les conditions de vie des personnes âgées, faciliteraient le maintien à domicile et apporteraient une aide aux personnes aidantes, proches et soignants.

Les principales contributions présentées dans cette thèse, à savoir la gestion de l'observabilité imparfaite en reconnaissance de plan par bibliothèque de plans, l'utilisation de filtres à particules, la conception d'une approche hybride temps réel, la reconnaissance d'anomalies dans une séquence d'actions en utilisant de la reconnaissance de plan et la reconnaissance d'objectif active, ont été publié dans des conférences scientifiques à comité de relecture. Nous avons choisi de publier dans *l'International Conference on Automated Planning and Scheduling* (ICAPS) car il s'agit d'une conférence spécialisée en planification, que nous trouvons réputée. De la même manière nous avons choisi *l'International Conference on Robotics*

and Automation (ICRA) pour sa spécialité en systèmes robotique, elle aussi que nous trouvons réputée.

Dans le premier chapitre, nous avons présenté une nouvelle approche à la reconnaissance de plan utilisant des plans hiérarchiques partiellement ordonnés, ainsi qu'une extension à la définition des bibliothèques de plans, permettant d'exprimer les différents types de bruit dans les observations. La nouvelle approche que nous proposons utilise des filtres à particules. Celle-ci consiste à simuler par construction descendante des arbres de plans possibles réalisés par un agent observé et de filtrer ces simulations en utilisant des observations afin d'avoir une population d'arbres. Ces arbres permettent une estimation des probabilités des objectifs de l'agent observé et de ses probables futures actions par dénombrement. Cette approche est à la fois temps réel, précise avec peu de ressources et offre une certaine robustesse au bruit dans les observations.

Dans le second chapitre, nous avons présenté PARC, un système intégrant à la fois de la reconnaissance de plan et de la reconnaissance d'activité. Dans PARC, nous représentons les actions de bas niveau par des interactions avec des objets, appelées des affordances. Ces affordances sont reconnues en détectant les mains et les objets dans les images envoyées par une caméra RGB-D en utilisant un RNN. Les affordances découvertes représentent les observations qui sont envoyées aux systèmes de reconnaissance de plan, un filtre à particule comme celui présenté au chapitre 1. À sa publication, PARC était un des premiers systèmes de reconnaissance de plan hybride temps réel fonctionnant sur un flux vidéo.

Dans le troisième chapitre, nous avons présenté deux approches de détection d'anomalie dans des séquences d'actions. La première de ces deux approches utilise les prédictions des objectifs obtenues par la reconnaissance de plan, combinées à un raisonnement bayésien, pour déterminer la probabilité qu'une action dans une sé-

quence soit normale ou une anomalie. La seconde approche utilise les prédictions des actions futures fournies par notre système de reconnaissance de plan par filtre à particules. Si une action a lieu alors que la probabilité d'occurrence prédite est sous un certain seuil, alors l'action est considérée comme anormale. Ces deux approches offrent des perspectives dans les applications de la reconnaissance de plan, comme par exemple en ciblant les interventions d'un assistant au moment d'apparitions des anomalies.

Dans ces trois premiers chapitres, nous avons utilisé des ensembles de données simulées similaires à ceux utilisés dans d'autres travaux sur le sujet. Ils ont été adaptés en fonction des problèmes pour correspondre aux situations étudiées, comme la présence d'observations bruitées, d'anomalies ou dans le cas d'un nombre restreint d'actions. Bien que simulés, ces ensembles de données illustrent des problèmes complexes permettant d'estimer les performances des approches présentées.

Dans le chapitre 4, nous avons présenté une approche active de la reconnaissance d'objectifs. La reconnaissance active d'objectif consiste à améliorer la prise des observations en planifiant de manière automatique en parallèle de la reconnaissance d'objectifs. Pour adresser ce problème, nous proposons une extension d'un *framework* de MOMDP existant, en rendant le problème de reconnaissance d'objectif explicite dans la représentation du modèle. Cette approche offre des débouchés pour l'amélioration des performances des capteurs avec des ressources limitées, comme les téléphones intelligents, et les performances des robots mobiles où l'angle d'observation impacte significativement les performances en termes de reconnaissance d'objectif.

En travaux futurs, afin d'appliquer ces approches à des problèmes réels, une des premières problématiques à adresser est la construction des bibliothèques de plans. L'apprentissage et la construction de celles-ci de manière automatique permet-

traient une application à plus grande échelle des techniques décrites dans cette thèse. Un second point d'amélioration se trouve dans la détection des activités. Dans PARC, les actions sont représentées par le contact entre une main et un objet sur un certain nombre d'images sur une certaine plage de temps. L'utilisation de techniques plus avancées en reconnaissance d'activité pourrait rendre notre système plus robuste sans pour autant trop sacrifier en termes de temps de calcul. Enfin une intégration de la reconnaissance d'objectif active aux autres approches décrites dans cette thèse serait une étape importante dans l'optique d'une intégration robot. Pour cela, il serait nécessaire d'étendre le modèle décrit aux plans hiérarchiques partiellement ordonnés, ce qui serait possible en utilisant des solveurs de POMDPs plus performants ou en séparant la planification des observations et la reconnaissance de plan.

RÉFÉRENCES

Aggarwal, J. K. et Ryoo, M. S. (2011). Human activity analysis : A review. *ACM Computing Surveys (CSUR)*, *43*(3), 16.

Amato, C. et Baisero, A. (2019). Active goal recognition. *arXiv preprint arXiv :1909.11173*.

Arora, A., Fiorino, H., Pellier, D., Métivier, M. et Pesty, S. (2018). A review of learning planning action models. *The Knowledge Engineering Review*, *33*.

Ashbrook, D. et Starner, T. (2002). Learning significant locations and predicting user movement with gps. Dans *International Symposium on Wearable Computers (ISWC)*, 101–108. IEEE.

Bandyopadhyay, T., Won, K. S., Frazzoli, E., Hsu, D., Lee, W. S. et Rus, D. (2013). Intention-aware motion planning. In *Algorithmic foundations of robotics X* 475–491. Springer.

Behnke, G., Höller, D. et Biundo, S. (2015). On the complexity of htn plan verification and its implications for plan recognition. Dans *International Conference on Automated Planning and Scheduling (ICAPS)*, 25–33.

Bisson, F., Larochelle, H. et Kabanza, F. (2015). Using a recursive neural network to learn an agent's decision model for plan recognition. Dans *International Joint Conference on Artifical Intelligence (IJCAI)*, 918–924.

Bouchard, B., Giroux, S. et Bouzouane, A. (2007). A keyhole plan recognition model for alzheimer's patients : First results. *Applied Artificial Intelligence*, *21*(7), 623–658.

Bui, H. H. (2003). A general model for online probabilistic plan recognition. Dans *International Joint Conference on Artifical Intelligence (IJCAI)*, 1309–1315.

Cappé, O., Godsill, S. J. et Moulines, E. (2007). An overview of existing methods and recent advances in sequential monte carlo. *Institute of Electrical and Electronics Engineers (IEEE)*, *95*(5), 899–924.

Charniak, E. et Goldman, R. P. (1993). A bayesian model of plan recognition. *Artificial Intelligence*, *64*(1), 53–79.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on information theory*, *2*(3), 113–124.

Del Moral, P. (1997). Nonlinear filtering : Interacting particle resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, *325*(6), 653–658.

Dutta, V. et Zielinska, T. (2018). Predicting human actions taking into account object affordances. *Journal of Intelligent & Robotic Systems*, 1–17.

Erol, K. (1996). *Hierarchical task network planning : formalization, analysis, and implementation*. (Thèse de doctorat).

Fahad, L. G. et Rajarajan, M. (2015). Anomalies detection in smart-home activities. Dans *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*, 419–422. IEEE.

Fern, A. et Tadepalli, P. (2010). A computational decision theory for interactive assistants. Dans *Advances in Neural Information Processing Systems (NIPS)*, 577–585.

Fortin-Simard, D., Bilodeau, J.-S., Bouchard, K., Gaboury, S., Bouchard, B. et Bouzouane, A. (2015). Exploiting passive rfid technology for activity recognition in smart homes. *IEEE Intelligent Systems*, *30*(4), 7–15.

Freedman, R. G. et Zilberstein, S. (2017). Integration of planning with recognition for responsive interaction using classical planners. Dans *AAAI Conference on Artificial Intelligence (AAAI)*.

Geib, C. (2009). Delaying commitment in plan recognition using combinatory categorial grammars. Dans *International Joint Conference on Artificial Intelligence (IJCAI)*.

Geib, C., Craensen, B. et Petrick, R. P. (2016). Combining plan recognition, goal reasoning, and planning for cooperative task behavior. Dans *International Joint Conference on Artificial Intelligence (IJCAI) Workshop on Goal Reasoning*, 1–8.

Geib, C. W. et Goldman, R. P. (2001). Plan recognition in intrusion detection systems. Dans *DARPA Information Survivability Conference (ISC)*, volume 1, 46–55. IEEE.

Geib, C. W. et Goldman, R. P. (2003). Recognizing plan/goal abandonment.

Dans *International Joint Conference on Artificial Intelligence (IJCAI)*, 1515–1517.

Geib, C. W. et Goldman, R. P. (2009). A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, *173*(11), 1101 – 1132.

Geib, C. W. et Kantharaju, P. (2018). Learning combinatory categorial grammars for plan recognition. Dans *Association for the Advancement of Artificial Intelligence (AAAI)*, 3007–3014.

Geib, C. W., Maraist, J. et Goldman, R. P. (2008). A new probabilistic plan recognition algorithm based on string rewriting. Dans *International Conference on Automated Planning and Scheduling (ICAPS)*, 91–98.

Ghallab, M., Nau, D. et Traverso, P. (2004). *Automated Planning : theory and practice*. Elsevier.

Gibson, J. J. (1979). The ecological approach to visual perception houghton mifflin. *New York*.

Giroux, S. et Pigot, H. (2012). Smart homes for people suffering from cognitive disorders. *Computer science and ambient intelligence (CSAI)*, 225–262.

Gmytrasiewicz, P. J. et Doshi, P. (2004). Interactive pomdps : Properties and preliminary results. Dans *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 1374–1375. IEEE Computer Society.

Granada, R. L., Pereira, R. F., Monteiro, J., Ruiz, D. D. A., Barros, R. C. et Meneguzzi, F. R. (2017). Hybrid activity and plan recognition for video streams. Dans *Plan, Activity and Intent Recognition Workshop - AAAI (AAAI-PAIR)*.

Hájek, P. (2013). *Metamathematics of fuzzy logic*, volume 4. Springer Science & Business Media.

Hochreiter, S. et Schmidhuber, J. (1997). Lstm can solve hard long time lag problems. Dans *Conference on Neural Information Processing Systems (NIPS)*, 473–479.

Hoffmann, J., Porteous, J. et Sebastia, L. (2004). Ordered landmarks in planning. *Journal of Artificial Intelligence Research*, *22*, 215–278.

Hu, J.-F., Zheng, W.-S., Lai, J. et Zhang, J. (2015). Jointly learning

heterogeneous features for rgb-d activity recognition. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5344–5352.

Jalal, A., Kamal, S. et Kim, D. (2015). Shape and motion features approach for activity tracking and recognition from kinect video camera. Dans *International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 445–450. IEEE.

Kabanza, F., Bellefeuille, P., Bisson, F., Benaskeur, A. R. et Irandoust, H. (2010). Opponent behaviour recognition for real-time strategy games. *AAAI workshop Plan, Activity, and Intent Recognition (AAAI-PAIR)*, *10*(5).

Kabanza, F., Filion, J., Benaskeur, A. R. et Irandoust, H. (2013). Controlling the hypothesis space in probabilistic plan recognition. Dans *International Joint Conference on Artifical Intelligence (IJCAI)*, 2306–2312.

Kantharaju, P., Ontañón, S. et Geib, C. W. (2019). Extracting ccgs for plan recognition in rts games. Dans *Association for the Advancement of Artificial Intelligence (AAAI)*.

Kautz, H. A. (1987). *A formal theory of plan recognition.* (Thèse de doctorat). Bell Laboratories.

Keller, T. et Eyerich, P. (2012). Prost : Probabilistic planning based on uct. Dans *International Conference on Automated Planning and Scheduling (ICAPS)*, volume 22.

Keren, S., Gal, A. et Karpas, E. (2014). Goal recognition design. Dans *International Conference on Automated Planning and Scheduling (ICAPS)*.

Kjellström, H., Romero, J. et Kragić, D. (2011). Visual object-action recognition : Inferring object affordances from human demonstration. *Computer Vision and Image Understanding*, *115*(1), 81–90.

Koppula, H. S., Gupta, R. et Saxena, A. (2013). Learning human activities and object affordances from rgb-d videos. *The International Journal of Robotics Research*, *32*(8), 951–970.

Koppula, H. S. et Saxena, A. (2016). Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence*, *38*(1), 14–29.

Kurniawati, H., Hsu, D. et Lee, W. S. (2008). Sarsop : Efficient point-based pomdp planning by approximating optimally reachable belief spaces. Dans *Robotics : Science and systems*, volume 2008. Zurich, Switzerland.

Laniel, S., Létourneau, D., Labbé, M., Grondin, F. et Michaud, F. (2017a). Enhancing a beam+ telepresence robot for remote home care applications. Dans *International Conference on Virtual Rehabilitation (ICVR)*, 1–2. IEEE.

Laniel, S., Létourneau, D., Labbé, M., Grondin, F., Polgar, J. et Michaud, F. (2017b). Adding navigation, artificial audition and vital sign monitoring capabilities to a telepresence mobile robot for remote home care applications. Dans *International Conference on Rehabilitation Robotics (ICORR)*, 809–811. IEEE.

Levine, S. J. et Williams, B. C. (2014). Concurrent plan recognition and execution for human-robot teams. Dans *International Conference on Automated Planning and Scheduling (ICAPS)*.

Liu, J. S. et Chen, R. (1998). Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association*, *93*(443), 1032–1044.

Lopes, M., Melo, F. S. et Montesano, L. (2007). Affordance-based imitation learning in robots. Dans *International Conference on Intelligent Robots and Systems (IROS)*, 1015–1021. IEEE.

Massardi, J. et Beaudry, É. (2017a). Activity and plan recognition applied to robot-assitant for daily living activities. Dans *Montreal AI Symposium*.

Massardi, J. et Beaudry, É. (2017b). Robot planning for activity recognition. Dans *Doctoral Consortium - International Conference on Automated Planning and Scheduling (ICAPS)*.

Massardi, J. et Beaudry, É. (2018). Robot planning for activity recognition. Dans *Doctoral Consortium - International Conference on Automated Planning and Scheduling (ICAPS)*.

Massardi, J. et Beaudry, É. (2020). Toward detecting anomalies in activities for daily living with a mobile robot using plan recognition. Dans *Robots for Health and Elderly Care - IROS (IROS-RobotHEC)*.

Massardi, J., Gravel, M. et Beaudry, E. (2019). Error-tolerant anytime approach to plan recognition using a particle filter. Dans *International Conference on Automated Planning and Scheduling (ICAPS)*, volume 29, 284–291.

Massardi, J., Gravel, M. et Beaudry, É. (2020). Parc : A plan and activity recognition component for assistive robots. Dans *IEEE International Conference on Robotics and Automation (ICRA)*, 3025–3031. IEEE.

Min, W., Mott, B. W., Rowe, J. P., Liu, B. et Lester, J. C. (2016). Player

goal recognition in open-world digital games with long short-term memory networks. Dans *International Joint Conference on Artificial Intelligence (IJCAI)*, 2590–2596.

Mirsky, R. et Gal, Y. (2016). Slim : Semi-lazy inference mechanism for plan recognition. Dans *International Joint Conference on Artificial Intelligence (IJCAI)*, 394–400. Récupéré de `http://dl.acm.org/citation.cfm?id=3060621.3060677`

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G. *et al.* (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529–533.

Montesano, L., Lopes, M., Bernardino, A. et Santos-Victor, J. (2008). Learning object affordances : from sensory–motor coordination to imitation. *IEEE Transactions on Robotics*, *24*(1), 15–26.

Peirce, C. S. (1903). *Pragmatism – The Logic of Abduction.*

Pereira, R. F., Oren, N. et Meneguzzi, F. (2020a). Landmark-based approaches for goal recognition as planning. *Artificial Intelligence*, *279*, 103217.

Pereira, R. F., Oren, N. et Meneguzzi, F. (2020b). Using sub-optimal plan detection to identify commitment abandonment in discrete environments. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *11*(2), 1–26.

Poppe, R. (2010). A survey on vision-based human action recognition. *Image and vision computing*, *28*(6), 976–990.

Rafferty, J., Nugent, C. D., Liu, J. et Chen, L. (2017). From activity recognition to intention recognition for assisted living within smart homes. *IEEE Transactions on Human-Machine Systems*, *47*(3), 368–379.

Rajbhandari, S., He, Y., Ruwase, O., Carbin, M. et Chilimbi, T. (2017). Optimizing cnns on multicores for scalability, performance and goodput. *ACM Special Interest Group on Operating Systems (SIGOPS)*, *51*(2), 267–280.

Ramırez, M. et Geffner, H. (2009). Plan recognition as planning. Dans *International Joint Conference on Artifical Intelligence (IJCAI)*, 1778–1783.

Ramırez, M. et Geffner, H. (2011). Goal recognition over pomdps : Inferring the intention of a pomdp agent. Dans *International Joint Conference on Artificial Intelligence (IJCAI)*, 2009–2014. Citeseer.

Redmon, J. et Farhadi, A. (2018). Yolov3 : An incremental improvement. *arXiv preprint arXiv :1804.02767*.

Revathi, A. et Kumar, D. (2017). An efficient system for anomaly detection using deep learning classifier. *Signal, Image and Video Processing*, *11*(2), 291–299.

Roy, P., Bouchard, B., Bouzouane, A. et Giroux, S. (2009). A hybrid plan recognition model for alzheimer's patients : interleaved-erroneous dilemma. *Web Intelligence and Agent Systems : An International Journal*, *7*(4), 375–397.

Shimada, A., Kondo, K., Deguchi, D., Morin, G. et Stern, H. (2013). Kitchen scene context based gesture recognition : A contest in icpr2012. In *Advances in Depth Image Analysis and applications (WDIA)* 168–185. Springer.

Shvo, M. et McIlraith, S. A. (2020). Active goal recognition. Dans *Association for the Advancement of Artificial Intelligence (AAAI)*, 9957–9966.

Sodemann, A. A., Ross, M. P. et Borghetti, B. J. (2012). A review of anomaly detection in automated surveillance. *IEEE Transactions on Systems, Man, and Cybernetics*, *42*(6), 1257–1272.

Somani, A., Ye, N., Hsu, D. et Lee, W. S. (2013). Despot : Online pomdp planning with regularization. Dans *Advances in Neural Information Processing Systems (NIPS)*, 1772–1780.

Song, D., Kyriazis, N., Oikonomidis, I., Papazov, C., Argyros, A. A., Burschka, D. et Kragic, D. (2013). Predicting human intention in visual observations of hand/object interactions. Dans *IEEE International Conference on Robotics and Automation (ICRA)*.

Sukthankar, G., Geib, C., Bui, H. H., Pynadath, D. et Goldman, R. P. (2014). *Plan, Activity, and Intent Recognition : Theory and practice*. Newnes.

Sultani, W., Chen, C. et Shah, M. (2018). Real-world anomaly detection in surveillance videos. Dans *IEEE Conference on Computer Vision and Pattern Recognition (CCVPR)*, 6479–6488.

Sung, J., Ponce, C., Selman, B. et Saxena, A. (2012). Unstructured human activity detection from rgbd images. Dans *IEEE International Conference on Robotics and Automation (ICRA)*, 842–849.

Vattam, S. S. et Aha, D. W. (2015a). Case-based plan recognition under imperfect observability. Dans *International Conference on Case-Based*

Reasoning (ICCBR), 381–395. Springer.

Vattam, S. S. et Aha, D. W. (2015b). Case-based plan recognition under imperfect observability. Dans *International Conference on Case-Based Reasoning (ICCBR)*, 381–395. Springer.

Vilain, M. B. (1991). Deduction as parsing : Tractable classification in the KL-ONE framework. Dans *Association for the Advancement of Artificial Intelligence (AAAI)*, 464–470.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Wu, J., Osuntogun, A., Choudhury, T., Philipose, M. et Rehg, J. M. (2007). A scalable approach to activity recognition based on object use. Dans *IEEE International Conference on Computer Vision (ICCV)*, 1–8. IEEE.

Xia, L., Gori, I., Aggarwal, J. K. et Ryoo, M. S. (2015). Robot-centric activity recognition from first-person rgb-d videos. Dans *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 357–364.

Yahaya, S. W., Lotfi, A. et Mahmud, M. (2019). A consensus novelty detection ensemble approach for anomaly detection in activities of daily living. *Applied Soft Computing*, *83*, 105613.

Zhu, Y., Nayak, N. M. et Roy-Chowdhury, A. K. (2012). Context-aware activity recognition and anomaly detection in video. *IEEE Journal of Selected Topics in Signal Processing*, *7*(1), 91–101.