UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

Conception d'un mécanisme intégré d'attention sélective dans une architecture comportementale pour robots autonomes

Thèse de doctorat
Spécialité : génie électrique

François FERLAND

Jury : Denis BÉLISLE
Yves LESPÉRANCE
Wael SULEIMAN (rapporteur)
François MICHAUD (directeur)

Sherbrooke (Québec) Canada Décembre 2014

À ma famille

RÉSUMÉ

Le vieillissement de la population à travers le monde nous amène à considérer sérieusement l'intégration dans notre quotidien de robots de service afin d'alléger les besoins pour la prestation de soins. Or, il n'existe pas présentement de robots de service suffisamment avancés pour être utiles en tant que véritables assistants à des personne en perte d'autonomie. Un des problèmes freinant le développement de tels robots en est un d'intégration logicielle. En effet, il est difficile d'intégrer les multiples capacités de perception et d'action nécessaires à interagir de manière naturelle et adéquate avec une personne en milieu réel, les limites des ressources de calculs disponibles sur une plateforme robotique étant rapidement atteintes.

Même si le cerveau humain a des capacités supérieures à un ordinateur, lui aussi a des limites sur ses capacités de traitement de l'information. Pour faire face à ces limites, l'humain gère ses capacités cognitives avec l'aide de l'attention sélective. L'attention sélective lui permet par exemple d'ignorer certains stimuli pour concentrer ses ressources sur ceux utiles à sa tâche. Puisque les robots pourraient grandement bénéficier d'un tel mécanisme, l'objectif de la thèse est de développer une architecture de contrôle intégrant un mécanisme d'attention sélective afin de diminuer la charge de calcul demandée par les différents modules de traitement du robot. L'architecture de contrôle utilisé est basée sur l'approche comportementale, et porte le nom HBBA, pour *Hybrid Behavior-Based Architecture*.

Pour répondre à cet objectif, le robot humanoïde nommé IRL-1 a été conçu pour permettre l'intégration de multiples capacités de perception et d'action sur une seule et même plateforme, afin de s'en servir comme plateforme expérimentale pouvant bénéficier de mécanismes d'attention sélective. Les capacités implémentées permettent d'interagir avec IRL-1 selon différentes modalités. IRL-1 peut être guidé physiquement en percevant les forces externes par le bias d'actionneurs élastiques utilisés dans la direction de sa plateforme omnidirectionnelle. La vision, le mouvement et l'audition ont été intégrés dans une interface de téléprésence augmentée. De plus, l'influence des délais de réaction à des sons dans l'environnement a pu être examinée. Cette implémentation a permis de valider l'usage de HBBA comme base de travail pour la prise de décision du robot, ainsi que d'explorer les limites en termes de capacités de traitement des modules sur le robot. Ensuite, un mécanisme d'attention sélective a été développé au sein de HBBA. Le mécanisme en question intègre l'activation de modules de traitement avec le filtrage perceptuel, soit la capacité de moduler la quantité de stimuli utilisés par les modules de traitement afin d'adapter le traitement aux ressources de calculs disponibles. Les résultats obtenus démontrent les bénéfices qu'apportent un tel mécanisme afin de permettre au robot d'optimiser l'usage de ses ressources de calculs afin de satisfaire ses buts.

De ces travaux résulte une base sur laquelle il est maintenant possible de poursuivre l'intégration de capacités encore plus avancées et ainsi progresser efficacement vers la conception de robots domestiques pouvant nous assister dans notre quotidien.

Mots-clés : Robotique, Architecture décisionnelle, Interactions humain-robot

REMERCIEMENTS

Tout d'abord, j'aimerais remercier mon directeur de thèse, François Michaud, pour son soutien constant.

Je désire remercier les Fonds de Recherche du Québec - Nature et Technologie et le Conseil de recherche en sciences naturelles et en génie du Canada pour leur support financier.

J'aimerais également remercier toute l'équipe de l'IntRoLab pour leur participation au développement à l'expérimentation des tests présentés aux chapitres 3 et 4, et particulièrement Dominic Létourneau pour son aide précieuse dans l'intégration matérielle et logicielle du robot IRL-1.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

CHAPITRE 1

Introduction

La science-fiction a habitué le public à une vision de la robotique qui a considérablement élevé ses attentes envers la technologie. Nous ne soupçonnons pas, par exemple, qu'un robot mobile puisse avoir de la difficulté à éviter des obstacles ou puisse s'égarer dans un environnement inconnu. Nous nous attendons plutôt à ce qu'il arrive à comprendre un langage naturel comme le français et qu'il soit capable de formuler des phrases complètes dans cette même langue. De plus, comme les robots industriels qui sont capables de précision et d'efficacité d'exécution sur les chaînes de montage, nous espérerons qu'un robot domestique présentera les mêmes caractéristiques en manipulant les objets courants d'une cuisine.

Ce genre de robot n'existe pas encore. Or, la science a fait des progrès fulgurants sur plusieurs facettes rattachées aux attentes envers un robot humanoïde sécuritaire pour des tâches domestiques ou de service. Il est donc permis de croire qu'en réunissant dans un même système les dernières découvertes en navigation autonome, en reconnaissance vocale et en manipulation compliante, par exemple, nous puissions obtenir un robot mobile véritablement utile et pouvant aider à des tâches de la vie courante.

Le problème en est ainsi un d'intégration et propose un défi de taille. En effet, les capacités à intégrer ont généralement été démontrées en isolation ou même en simulation en temps différé, c.-à-d., avec une capacité de calcul virtuellement illimitée. Si un sous-système de navigation autonome est efficace seulement lorsqu'il nécessite la presque totalité des ressources d'un processeur moderne, il n'est pas garanti qu'un robot tentant de l'exploiter simultanément avec d'autres sous-systèmes tout aussi gourmands puisse le faire de manière satisfaisante. De plus, il y a toujours une limite à l'optimisation de ces systèmes. Nous souhaitons aussi réutiliser sans modifications le plus grand nombre de composants déjà disponibles et ainsi bénéficier des efforts investis par d'autres groupes de recherche à travers le monde. Nous devons donc imaginer d'autres solutions.

Sur ce point, nous pouvons nous inspirer de nos propres processus cognitifs en tant qu'êtres humains. Nous sommes tous conscients que nous ne pouvons pas exécuter simultanément plusieurs tâches demandant une forte charge cognitive. Parfois, une forte charge perceptuelle peut même nuire à l'accomplissement d'une seule tâche en réduisant notre concen-

tration. Des travaux en psychologie s'intéressent à nos limitations lorsque nous faisons face à un trop grand nombre de stimuli et de la façon dont nous orientons notre attention. Lavie et Tsal [1994] en relèvent plusieurs, tels que les travaux de Eriksen et Hoffman [1972, 1973] qui étudient l'effet d'éléments distracteurs sur l'identification d'éléments visuels. Ils montrent que les délais de réponse des participants sont plus longs si les éléments distracteurs et les éléments à identifier n'appartiennent pas à la même catégorie, et que l'effet d'interférence des distracteurs s'estompe s'ils sont suffisamment loins de la cible à identifier. Ce genre de découverte implique que nous pouvons focaliser notre attention visuelle pour ignorer une partie des stimuli qui ne sont pas près de notre zone de travail.

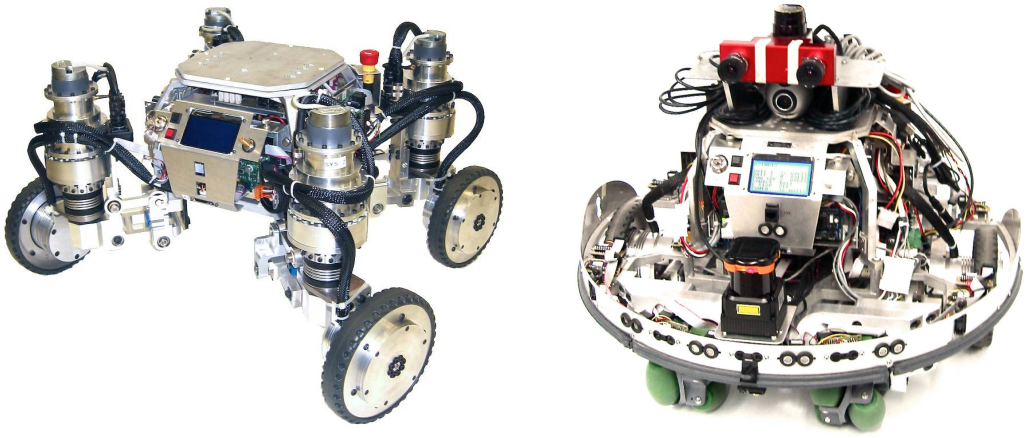
En robotique, des groupes de recherche se sont concentrés sur la reproduction du phénomène de l'attention, particulièrement au niveau de la vision artificielle, en dirigeant par exemple le regard vers les zones visuellement actives d'une scène [Breazeal et Scassellati, 1999; Ruesch *et al.*, 2008]. De plus, certains environnements de développement en robotique tels que OROCOS [Bruyninckx, 2001] ou ROS [Quigley *et al.*, 2009] permettent de distribuer plusieurs processus cognitifs évoluant en parallèle. Or, ces environnements ne permettent pas nécessairement de moduler facilement l'exécution de sous-modules selon l'attention que devrait porter le robot à son environnement en fonction du contexte et les tâches que nous souhaitons le voir accomplir.

Aborder le sujet de l'attention sélective pour un robot humanoïde implique la poursuite de deux objectifs principaux :

- La conception d'un torse humanoïde, IRL-1, permettant l'intégration de plusieurs capacités sensorielles et motrices. IRL-1 est doté d'une tête expressive articulée intégrant un capteur 3D Kinect de Microsoft, d'un caméra panoramique, de 8 microphones, de deux bras compliants à quatre degrés de liberté et deux pinces. Il peut s'installer sur deux plateformes mobiles, AZIMUT-3 [Ferland *et al.*, 2010] et Telerobot [Michaud *et al.*, 2010a], montrées à la figure 1.1. Il en résulte une plateforme robotique humanoïde et interactive qui permet de réaliser une grande variété d'expérimentation dans des environnements réels et dynamiques. La mobilité de la plateforme permet d'intégrer autant des capacités de cartographie et de localisation simultanées que des techniques de navigation autonome. La compliance des actionneurs d'IRL-1 permet la mise en place de comportements d'interaction directe avec les usagers, de façon naturelle et sécuritaire. Par exemple, grâce au contrôle en couple permis par les actionneurs compliants sur AZIMUT-3 ou dans les bras d'IRL-1, il est possible de guider le robot en appliquant une force directement sur le robot, sans craindre de se blesser ou d'abimer les moteurs. L'ensemble de capteurs

sur le robot permet également d'évaluer un vaste éventail d'algorithmes de perception pour la détection et la localisation d'objets et de personnes, par des approches visuelles ou sonores. Ces algorithmes peuvent être mis en œuvre dans des scénarios d'applications allant de la téléprésence robotique à des déploiements complètement autonomes. Ainsi, en ayant accès à une plateforme robotique aux capacités multiples, il est possible de créer des conditions expérimentales permettant d'étudier l'attention sélective dans des contextes réels et pragmatiques visant à optimiser l'usage des ressources à la disposition du robot.

- Le développement d'une architecture décisionnelle permettant d'agencer ces capacités en fonction des ressources de calculs du robot. Cette architecture, nommée HBBA (pour *Hybrid Behavior-Based Architecture*) [Michaud *et al.*, 2010b], est une évolution directe de MBA (pour *Motivational Behavioral Architecture*) [Michaud *et al.*, 2007a]. La majorité des nouveautés se retrouvent dans le système perceptuel haut-niveau et le mécanisme d'attention sélective intégré à un module s'appelant l'*Intention Workspace*. Ce mécanisme d'attention sélective consiste à venir activer dynamiquement les comportements [Mataric et Michaud, 2008] que le robot doit utiliser pour l'atteinte de ses objectifs, rôle que joue l'*Intention Workspace*. Le nouveau mécanisme intégré de façon harmonieuse à l'*Intention Workspace* est basée sur le filtrage perceptuel décrit par Treisman [1960, 1964], qui implique que certains stimuli à des modules de traitement peuvent être modulés afin de limiter la charge de calculs. Pour valider l'approche, une implémentation de l'architecture à l'aide de l'infrastructure ROS (pour *Robot Operating System*) [Quigley *et al.*, 2009] a été réalisée.



(a) AZIMUT-3 (b) Telerobot

Figure 1.1 Les plateformes mobiles disponibles pour IRL-1.

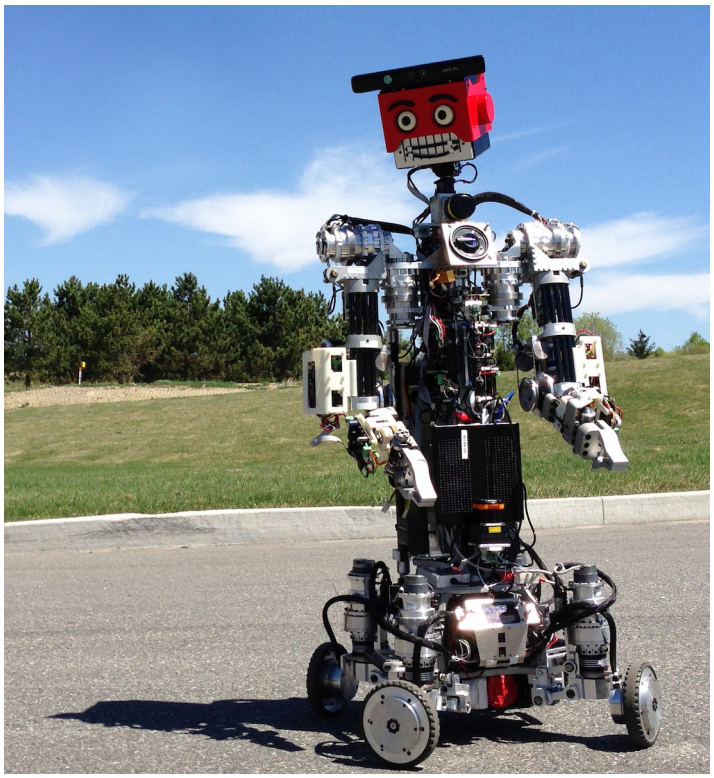


Figure 1.2 IRL-1 en configuration AZ3.

L'organisation de la thèse correspond aux travaux réalisés pour atteindre ces deux objectifs. Tout d'abord, le chapitre 2 présente une revue de littérature sur le phénomène de l'attention sélective et des architectures de contrôle en robotique. Le chapitre 3 présente l'article de journal publié dans la revue *Journal of Human-Robot Interaction* [Ferland *et al.*, 2012] qui décrit la conception de IRL-1 ainsi que trois tests préliminaires venant évaluer deux techniques d'interaction ainsi qu'un premier aspect du filtrage perceptuel de HBBA. Le chapitre 4 correspond à un second article de journal, soumis à la revue *Autonomous Robots* [Ferland et Michaud, 2014], qui décrit en détail les mécanismes d'attention sélective intégrant l'activation de modules et le filtrage perceptuel, en plus de valider son impact positif sur la répartition des charges de calculs d'IRL-1 en fonction des tâches à réaliser. Finalement, le chapitre 5 conclut la thèse en faisant ressortir les contributions scientifiques apportées ainsi qu'en décrivant une suite envisageable pour les travaux.

CHAPITRE 2

Attention et architectures de contrôle

Ce chapitre rassemble les notions d'attention sélective et les architectures de contrôle, car l'objectif fondamental d'une architecture de contrôle est de coordonner l'usage des modules de traitement de l'information par le robot afin d'atteindre ses buts et de réaliser les tâches qui lui sont assignées. Réaliser cette coordination implique incidemment des mécanismes d'attention sélective. Dans le présent chapitre, le phénomène de l'attention en psychologie est examiné afin de situer comment ceci est pris en considération dans les architectures de contrôle en robotique.

2.1 Phénomène de l'attention en psychologie

Le phénomène de l'attention chez l'humain peut être une source importante d'inspiration dans le développement d'un système robotique intégrant plusieurs capacités distinctes. Cette section présente des travaux effectués en psychologie qui permettent de justifier l'intégration de l'attention dans une architecture de contrôle, non seulement dans le but de reproduire des caractéristiques humaines, mais surtout dans le but d'allouer efficacement la capacité de traitement aux informations perceptuelles disponibles.

2.1.1 Attention sélective

Le problème de sélection de l'attention est étudié depuis longtemps en psychologie et est la source de multiples débats. Lavie et Tsal [1994] en font une analyse intéressante et Bundesen et Habekost [2004] présentent un bon résumé de la situation actuelle.

Broadbent [1958] a présenté les bases de la théorie de la **sélection hâtive** qui propose que nos capacités perceptuelles sont limitées et qu'un tri des stimuli doit être effectué très tôt dans la chaîne de traitement, ce qui entraîne une mise à l'écart complète des stimuli jugés non-significatifs. À l'opposé, Deutsch et Deutsch [1963, 1967] et Norman [1968] supportent plutôt une **sélection tardive**, qui suppose que notre perception serait illimitée, effectuée de façon automatique et parallèle, et que la sélection s'opère seulement après une perception complète de tous les stimuli. Selon ce modèle, l'attention serait plutôt une faculté de nos mécanismes de réponse que de nos capacités perceptuelles. Comme tous les stimuli

sont traités, ils doivent être jugés selon leur pertinence. Pour que leur pertinence soit jugée, leur contenu doit nécessairement être analysé. L'attention se manifesterait ensuite par quels percepts ont été choisis comme étant importants.

Dans le camp de la sélection hâtive, Treisman et Geffen [1967] ont tenté par la suite de vérifier l'inverse, c.-à-d., à quel point l'attention était une caractéristique de la perception plutôt que de notre mécanisme de réponse, à partir d'une expérience sur l'analyse verbale de deux messages concurrents, un primaire et l'autre secondaire, exigeant deux réponses différentes. L'objectif de cette expérience était de comparer la compétition de la perception et de la réponse en écoute sélective, mais aussi d'explorer en plus amples détails la nature de la limite de l'attention sélective. Les résultats de l'expérience favorisent nettement la sélection au niveau perceptuel. Les auteurs citent d'autres travaux (de Treisman [1960, 1964], basés sur ceux de Tanner et Swets [1954]) quant à la nature du filtre perceptuel. Celui-ci ne supprimerait pas entièrement l'analyse du contenu des messages secondaires, mais réduirait plutôt leur importance (leur rapport signal sur bruit), ce qui expliquerait pourquoi certains mots secondaires peuvent tout de même être perçus. Ceci permettrait donc le monitorage ou la surveillance de stimuli secondaires en cas d'apparition d'événements importants. Cet effet d'atténuation du message secondaire semble être directement affecté par l'attention portée au message principal. En effet, les auteurs n'ont remarqué aucune corrélation entre l'expérience faisant appel à l'attention et une autre masquant certaines parties du message par un bruit extérieur, laissant croire que le phénomène d'atténuation du message secondaire serait variable.

La proposition de Lavie [1995] se situe à mi-chemin entre les deux paradigmes. Elle avance que nos capacités perceptuelles sont effectivement limitées, mais que l'allocation se fait de façon automatique jusqu'à épuisement des ressources selon la priorité accordée aux stimuli en fonction de l'attention qui leur est portée. Ainsi, les stimuli non-significatifs ne seraient ignorés que lorsque ceux significatifs utilisent la totalité des ressources disponibles. En fait, la charge perceptuelle aurait un impact direct sur l'attention. Une forte charge serait le seul moyen d'enclencher une sélection hâtive. Un indice de la validité de cette théorie a été présenté par Lavie et Fox [2000]. Il est basé sur le phénomène de l'amorçage négatif (*negative priming* [Tipper, 1985]), qui correspond au ralentissement de la réponse à des stimuli ignorés par le passé [Bundesen et Habekost, 2004] et est généralement considéré comme une preuve de l'existence de la sélection tardive, puisqu'elle nécessite une reconnaissance du stimulus avant sa réjection. L'expérience montre que ce phénomène diminue avec l'augmentation de la charge perceptuelle, ce qui permet de supposer qu'une sélection hâtive prend place. Finalement, le type de stimuli affecterait également le traitement de distrac-

teurs pour une sélection tardive [Lavie, 2005]. L'existence d'une limitation en ressources perceptuelles est bien observée en psychologie, mais sa source exacte reste à déterminer en neurologie, selon la même auteure. En robotique, nous connaissons plutôt bien la nature de cette limitation. Nous devrons cependant tenir compte autant de l'importance que du type de la charge perceptuelle dans l'allocation des ressources du robot.

2.1.2 Reproductions de mécanismes d'attention en robotique

Le phénomène de l'attention est un sujet déjà populaire en robotique. Or, s'il est possible de trouver plusieurs travaux concernant la reproduction de mécanismes d'attention pour orienter efficacement les ressources motrices limitées d'une plateforme robotique (par exemple, on ne peut pas regarder à deux endroits en même temps), il est plutôt rare de trouver des travaux qui tentent d'effectuer une attention sélective d'un point de vue perceptuel et cognitif. Pour reprendre la taxonomie de la section 2.1.1, la robotique actuelle suivrait ainsi le modèle de la sélection tardive plutôt qu'hâtive. Paletta *et al.* [2005] recensent plusieurs approches en vision artificielle appliquées à la robotique et qui mettent en œuvre des mécanismes d'attention servant à orienter l'action. Par exemple, Vijayakumar *et al.* [2001] proposent un système d'orientation de la tête et des yeux d'un robot humanoïde vers l'élément le plus pertinent d'une scène. Cet élément est déterminé à partir d'une carte de relief (*saliency*) générée par un réseau de neurones.

Les travaux de Breazeal et Scassellati [1999] avec le robot *Kismet* (voir figure 2.1) démontrent l'intégration d'un système d'attention visuel dans une architecture comportementale incluant des sources motivationnelles pour diriger le regard selon le contexte.

Basé sur le modèle de Wolfe [1994], lui-même inspiré par les travaux de Treisman [1985] sur le traitement visuel préattentif, Kismet utilise un système visuel à deux niveaux. Le premier, suggéré par Neisser [1967], est massivement parallèlisé et traite toute l'image pour la détection de caractéristiques de base comme la couleur et le mouvement. Le deuxième, limité en ressources et sélectif, dirige les fonctions cognitives comme la reconnaissance de visages à des sections limitées de l'image. Le système développé pour *Kismet* génère plusieurs cartes de caractéristiques qui indiquent leur présence par un niveau de 8-bits pour chaque pixel de l'image. Ces cartes de caractéristiques sont ensuite combinées pour produire une carte d'activation de l'attention. Cette carte est ensuite segmentée en régions similaires pour que les trois régions les plus importantes en taille soient acheminées au contrôle moteur des yeux et aux systèmes comportementaux et motivationels. Dépendemment du contexte, le système module le gain appliqué aux cartes des caractéristiques. Le contexte est dérivé de la motivation actuelle du robot, qui influence l'activation des

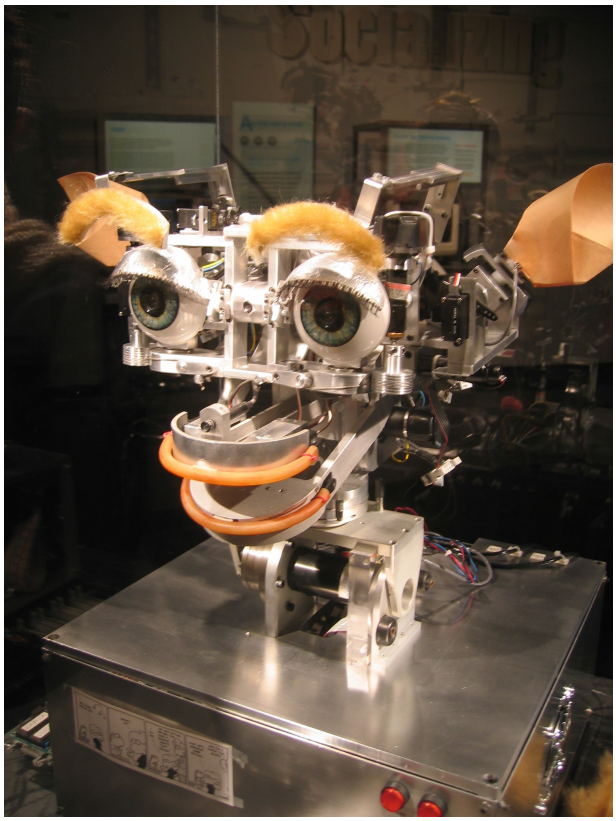


Figure 2.1 Le robot *Kismet* [Breazeal et Scassellati, 1999].

comportements. Par exemple, si le comportement de recherche de personnes est activé, la carte de recherche de visages aura plus de poids dans la carte d'activation combinée, ce qui devrait ensuite influencer l'orientation du regard de la plateforme. Les stimuli différents, comme la couleur ou le mouvement, demeurent donc traités mais éventuellement ignorés par le mécanisme d'attention.

L'analyse par Ude *et al.* [2007] de *Kismet* indique que l'approche développée par Breazeal et Scassellati, même si le but actuel influence la sélection de l'attention, serait principalement basée sur les données entrantes en les acheminant du bas vers le haut, c.-à-d., strictement des capteurs vers les processus cognitifs plus élevés, sans permettre un retour vers les couches plus primitives pour moduler le traitement. Ce que Ude *et al.* [2007] proposent comme nouvelle approche implique également une influence des buts, donc du haut vers le bas, en intégrant un mécanisme de modification des gains lors de la combinaison de plusieurs cartes de caractéristiques en une seule carte de relief. Or, ceci est plutôt similaire à la façon dont les motivations influencent justement les gains de combinaison dans le système décrit par Breazeal et Scassellati. Il pourrait être argumenté que, puisque les motivations sont également affectées par les stimuli, *Kismet* demeure basée principalement sur les données entrantes, mais nous croyons qu'il est plus juste de considérer le système

attentionel de *Kismet* comme étant bidirectionnel, c.-à-d., que des signaux circulent autant des couches inférieures vers les couches supérieures que des couches supérieures vers les couches inférieures pour mettre en œuvre le mécanisme d'attention.

Plus récemment, Ruesch *et al.* [2008] présentaient une implémentation pour l'*iCub* [Tsagarakis *et al.*, 2007] d'un système d'attention visuelle et auditive. Le système, distribué, multimodal et strictement bas vers le haut, projette sur une sphère entourant la tête du robot une carte de relief multimodale. Puisque la représentation est égocentrique, la surface de projection est appellée egosphère [Albus, 1991]. L'implémentation est disponible sous forme de logiciel libre et fonctionne sur des ordinateurs conventionnels plutôt qu'un ensemble de processeurs spécialisés comme pour *Kismet*, ce qui est tout à fait normal considérant qu'il y a presque dix ans qui séparent les deux implémentations.

Les mécanismes d'attention peuvent également être utilisés pour s'extirper d'une situation problématique. Dans [Scheutz et Andronache, 2004], une expérimentation place un robot mobile où certains de ses comportements par leur compétition l'empêchent de progresser vers une cible. Le problème provient principalement d'un comportement tentant d'atteindre une cible à partir de la reconnaissance d'un objet dans un flux vidéo. Dès que l'objet en question est détecté, le comportement dirige le robot vers un passage trop étroit pour lui et le comportement d'évitement d'obstacle le force à faire demi-tour. Le comportement global du robot devient donc oscillatoire et entre dans une boucle sans fin. Quand le mécanisme de supervision détecte l'absence de progrès, il coupe le flux vidéo, ce qui empêche le comportement d'atteinte de cible d'agir, et permet au reste du système décisionnel de contourner l'obstacle. L'attention du robot est donc redirigée. Lorsque la cible réapparaît, le mécanisme de supervision rétablit le flux vidéo et donc l'attention du robot vers la cible.

Effet sur l'allocation des ressources de calcul

Breazeal et Scassellati [1999] prétendent également que leur système d'attention permet de diriger les ressources de calcul selon le contexte, mais en donnent malheureusement peu de détails. Cet aspect du système semble être limité à l'activation et à la désactivation des comportements selon le contexte, puisque les algorithmes de détection visuelle fonctionnent en tout temps et sur toute l'image, ce qui correspond à la suggestion de Neisser [1967]. Il est permis de supposer que le fait de réorienter le regard éloigne l'attention des éléments distracteurs, ce qui pourrait diminuer le traitement nécessaire aux stimuli, mais ceci n'est pas mentionné dans leurs travaux.

L'architecture HAMMER (*Hierarchical Attentive Multiple Models for Execution and Recognition*) [Demiris et Khadhouri, 2006] considère également les ressources limitées d'une plateforme robotique dans le cadre d'un modèle d'attention visuelle. L'architecture est composée de plusieurs paires de modèles inverses et directs. Le modèle inverse commande un ou plusieurs actionneurs au mieux de ses connaissances selon le but à atteindre. Le modèle direct est prédictif et procure un estimé de l'état futur au modèle inverse. L'accès aux ressources est formulé comme des requêtes à un mécanisme d'attention. Le mécanisme contrôle l'exécution des paires de modèles en fonction du niveau de confiance des modèles inverses, les plus confiants étant prioritaires. Par exemple, le niveau de confiance dans une tâche de manipulation peut être liée à la distance entre l'outil et l'objet à manipuler : Plus la cible est près de l'outil, plus le modèle est confiant de ses actions, et plus l'attention favorisera l'exécution de la paire de modèles associée à cette tâche.

Le besoin d'allouer efficacement les ressources d'un robot ou d'un agent intelligent a été soulevé plusieurs fois. Le besoin de limiter à un sous-ensemble l'information perceptuelle dans le but d'améliorer les performances générales du système a été proposé par Scheutz et Andronache [2004], et le manque de ressources pour traiter simultanément l'ensemble des stimuli, notamment visuels et auditifs, est un thème récurrent du projet *Spartacus* [Michaud *et al.*, 2007a]. Plus récemment, Langley *et al.* [2009] identifiaient la nécessité de développer des architectures permettant d'allouer les ressources d'un agent pour bien diriger son attention perceptuelle, ses actionneurs et la tâche qu'il poursuit. Paletta *et al.* [2005] soulignent le besoin d'utiliser l'attention sélective non seulement pour traiter l'information pertinente à une tâche, mais aussi pour choisir les sources de données appropriées pour la réponse à un état spécifique du système dans le temps et l'espace.

Le *CoSy Architecture Schema* (CAS) [Hawes *et al.*, 2007] et son infrastructure CAST (pour CAS *Toolkit*) [Hawes et Wyatt, 2010] est une architecture de contrôle récente qui permet un ajustement dynamique des ressources en calcul. Il divise ses processus en deux catégories : gérés (*managed*) et non-gérés (*unmanaged*). Les processus non-gérés sont ceux qui ne demandent pas de puissance de traitement significative et peuvent donc être actifs en tout temps. Ils sont décrits comme préattentifs. À l'inverse, les processus gérés sont suggérés et activés selon les besoins du système. Ces processus, appelés aussi sous-modules, sont divisés en sous-architectures ou multiples espaces de travail partagés (*Multiple Shared Workspaces*). Ceci permet plusieurs représentations différentes et distribuées des stimuli. Chaque sous-architecture possède une mémoire de travail, fonctionnant sur le principe d'une structure de données associative permettant de récupérer une information à partir d'une clé unique. Pour communiquer entre elles, les sous-architectures peuvent accéder

en lecture et en écriture à ces mémoires de travail. Tout changement à une mémoire de travail est appelé un événement, et chaque événement est diffusé à toutes les mémoires de travail de toutes les sous-architectures, et donc à tous les sous-modules qui les composent. La production d'un événement a donc un impact sur la charge de calcul du système. En contrôlant quels événements parviennent aux architectures, le CAST peut donc limiter la charge de calcul, et ainsi mettre en œuvre une forme de filtrage perceptuel sur le type d'information circulant entre les sous-architectures. L'impact de traiter des événements superflus par les différents sous-modules a également été mesuré en fonction du nombre de sous-architectures [Hawes *et al.*, 2009b]. Notons qu'ici le coût de traitement d'un événement ne représente pas le traitement de l'information associée à cet événement, mais seulement de l'acheminer aux sous-modules des sous-architectures. Avec CAST, comme le filtrage n'est possible qu'entre les sous-architectures, la production d'un événement a donc toujours un coût de traitement selon le nombre de sous-modules présents dans une même sous-architecture. L'étude de l'impact conclut qu'il y a un équilibre à trouver entre l'inclusion de tous les sous-modules de traitement dans une seule et même sous-architecture, où le filtrage est impossible, et avoir chaque sous-module dans sa propre sous-architecture, où le coût associé à la communication entre sous-architectures devient plus important que celui épargné par le filtrage.

Wyatt et Hawes [2008] indiquent qu'une architecture comportant plusieurs modalités perceptuelles et d'actions distribuées tout en ayant des ressources de calcul limitées fait face à au moins quatre problèmes :

- **Le filtrage :** À quel(s) module(s) envoyer l'information ? À ne pas confondre avec la notion de filtre perceptuel de Treisman [1960, 1964]. Wyatt et Hawes [2008] parlent plutôt d'aiguillage.

- **La liaison d'information :** Comment combiner plusieurs sources d'informations correspondant à un même événement ?

- **La fusion d'action :** Comment combiner plusieurs commandes motrices ?

- **La gestion du traitement :** Comment allouer les ressources de la plateforme selon le contexte ?

Pour ce dernier problème, qui nous intéresse plus particulièrement dans cette section, les auteurs donnent un exemple intéressant : si nous regardons une tasse dans le but de la reconnaître, nous retirons des informations différentes que lorsque nous la regardons

dans le but de la prendre. Il apparaît donc important de considérer le but en cours dans l'allocation de nos ressources.

Leur approche est décrite plus en détails avec PECAS (*PlayMate/Explorer CoSy Architecture Sub-Schema* [Hawes *et al.*, 2009c]), une spécialisation du CAS pour deux robots, *PlayMate* et *Explorer*. Les processus gérés proposent en fait des objectifs à atteindre pour le robot. Certains de ces objectifs sont purements épistémiques, c.-à-d., ils visent l'apprentissage d'une nouvelle connaissance. Par exemple, une sous-architecture de planification pourrait vouloir interroger une sous-architecture de navigation pour obtenir une carte globale à jour. Chaque sous-architecture est considérée comme un agent, et l'allocation des ressources devient un problème de planification multiagent. La sous-architecture de planification, en partant des tâches à accomplir, peut donc gérer le flot d'information à travers le système. Le flot d'information est géré par la configuration de filtres d'événements du CAST présents entre les mémoires de travail des sous-architectures Dans leurs termes, ceci permet de focaliser l'attention à un sous-ensemble d'informations plutôt que de surcharger une seule mémoire de travail globale avec tous les stimuli disponibles.

2.2 Architectures - Terminologie

En robotique, le terme **architecture** peut référer à deux concepts : la structure d'un système robotique, c.-à-d., ses sous-systèmes et leurs interactions, et le style architectural qui soutend la structure selon un ensemble de concepts informatiques, p. ex., une approche client-serveur [Kortenkamp et Simmons, 2008]. En informatique, le terme **architecture logicielle** (*software architecture*) peut, pour un programme ou un système informatique, représenter la (ou les) structure(s) du système, qui comprend des éléments logiciels, les propriétés extérieurement visibles de ces éléments et les relations entre ces éléments [Bass *et al.*, 2003]. Cette définition est donc similaire aux concepts nommés par Kortenkamp et Simmons. En intelligence artificielle, le terme **architecture cognitive** est souvent utilisé pour spécifier l'infrastructure supportant un système intelligent [Langley *et al.*, 2009]. Un tel système peut exister exclusivement en simulation ou être incarné (*embodied*) par un robot mobile et situé (*situated*) dans son environnement, c.-à-d., il l'expérimente directement plutôt que par l'entremise de représentations abstraites [Brooks, 1991].

La terminologie des architectures a également été abordée par Hawes et Wyatt [2010], qui décrivent trois niveaux de description de la conception d'une architecture. Le premier niveau, appelé *computational architecture* ou *information-processing architecture* décrit une structure de traitement de l'information sans nécessairement expliquer quel problème elle

tente de régler. Il s'agit donc du niveau d'abstraction le plus élevé pour la description d'une architecture. Le deuxième niveau, appelé *instanciated information-processing architecture*, reprend la structure décrite au premier niveau, mais le problème de traitement d'information y est maintenant bien défini. Le troisième et dernier niveau est appelé *software architecture* et vient décrire de quelle façon est implémentée l'architecture.

Même illustrées, ces différentes conceptions du terme architecture peuvent être facilement confondues. Il est donc nécessaire d'être constants dans notre utilisation de ces termes. En robotique, le terme **architecture de contrôle** est souvent utilisé pour désigner la façon dont les systèmes sensoriels et moteurs sont liés dans le but d'obtenir un comportement spécifique [Pfeifer et Scheier, 1999]. Le terme **architecture décisionnelle** est également utilisé à cet effet, en admettant qu'un véritable processus de décision ait lieu. Dans les deux cas, ceci correspond généralement au premier ou au deuxième niveau de Hawes et Wyatt [2010], selon le niveau de détails présentés dans la description de l'architecture.

Pour cette thèse, nous utilisons **architecture de contrôle** pour deux raisons : il est plus général, même si une prise de décision a nécessairement lieu dans les systèmes qui nous intéressent, et le terme est plus près de son équivalement anglophone (*control architecture*), abondamment présent dans la littérature. Le terme **architecture cognitive** est réservé pour décrire des systèmes ou des sous-systèmes possèdant des capacités cognitives plus évoluées et généralement inspirées de l'intelligence humaine. Ces systèmes ou sous-systèmes manipulent généralement des symboles ou des représentations abstraites de l'environnement. Par exemple, un tel système ne traitera pas directement l'image d'une caméra vidéo mais réagira à un événement simplifié comme la détection d'un visage. Ce type d'architecture peut également être utilisé pour interagir avec un planificateur de tâche qui fonctionne également avec des descriptions abstraites d'actions plutôt que des commandes motrices. Finalement, lorsque nous parlons d'implémenter une architecture, nous signifions la réalisation d'un système selon une architecture particulière. Les termes **infrastructure** et *framework* sont souvent utilisés pour décrire ce qui soutient un tel système. Celui-ci correspond nécessairement au troisième niveau décrit par Hawes et Wyatt [2010].

Une façon simple de représenter une architecture de contrôle par rapport à son environnement est présentée à la figure 2.2. Les capteurs permettent de mesurer l'environnement et les actionneurs d'agir sur lui. Peu importe la forme de l'architecture de contrôle et à moins de vouloir se priver d'action ou de perception, il est impossible de se libérer de cette chaîne qui lie le robot à son environnement.

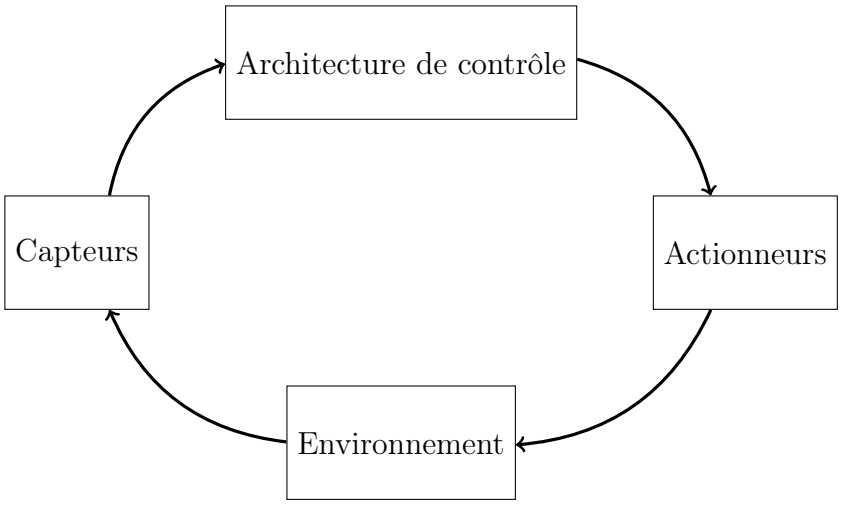


Figure 2.2 Structure de base des interactions entre un robot et son environnement.

2.2.1 Évolution des architectures de contrôle

Shakey [Nilsson, 1969] a été le premier d'une longue série de robots mobiles à exploiter une architecture de contrôle purement délibérative, ou *Think then Act* [Mataric *et al.*, 2002], qui implique une séparation temporelle stricte entre la perception de l'environnement, l'élaboration d'un plan de contrôle et le suivi de ce dernier. Ces architectures délibératives ont été critiquées pour leur manque d'adaptabilité dans des scénarios réels et leur impossibilité à réagir rapidement à des événements dynamiques [Brooks, 1990].

L'approche comportementale développée dans les années 1980 s'apparente à la modalité de *Think the Way You Act* [Mataric *et al.*, 2002]. S'inspirant de la biologie, cette approche repose sur l'émergence de comportements évoluées à partir des interactions de comportements plus primitifs. Avec l'architecture par priorités, Brooks [1986] décrit une façon simple et efficace de coordonner différentes actions. L'architecture met en œuvre plusieurs processus décisionnels concurrents, appelés comportements, qui compétitionnent pour l'accès aux actionneurs. Les comportements sont classés par priorités, les plus hauts inhibant les actions des plus bas. C'est ce qu'on appelle généralement l'arbitration des comportements, ou sélection d'action [Pirjanian, 2000]. Chaque comportement demeure simple et permet l'exécution d'une seule tâche de façon indépendante, comme éviter des obstacles ou se rendre à une base de recharge. Avec les schémas moteurs, Arkin [1998] propose une forme d'arbitration qui, plutôt que de faire agir un seul comportement à la fois, permet de fusionner l'effet de plusieurs comportements concurrents. Plus récemment, Antonelli *et al.* [2005] ont proposé une technique mathématique basée sur l'espace nul d'une ma-

trice de contrôle permettant la généralisation, du moins en évitement d'obstacle, des deux formes d'arbitration mentionnées plus haut en plus de pouvoir générer des trajectoires plus souples et naturelles.

Si les architectures comportementales classiques ne comportent pas de planificateur à long terme ou de représentations centralisées, les architectures hybrides permettent de combiner plusieurs aspects des approches réactives et délibératives, ou *Think and Act Independently in Parallel* [Mataric *et al.*, 2002]. Ces architectures incluent généralement un module exécutif qui gère les conflits entre les deux couches opérant selon des échelles de temps différentes. C'est pour cela qu'elles sont appelées architectures à trois niveaux. Des architectures comportementales multiniveaux permettent l'introduction de facultées cognitives plus évoluées comme la planification de tâches. MBA [Michaud *et al.*, 2007a], présentée à la section 2.2.3, en est un bon exemple. La grande différence entre les approches hybride et comportementale multiniveau réside dans l'emplacement de la représentation de l'environnement. Elle se retrouve centralisée dans la partie délibérative d'une architecture hybride alors qu'elle demeure distribuée à travers les différents modules comportementaux pour l'autre approche. Finalement, ces deux approches peuvent être liées à des observations en psychologie, notamment avec le modèle de Norman et Shallice [1986] qui avance l'existence de processus automatiques et délibérés régulés par l'attention en cognition humaine. Des expériences en psychologie [Gazzaniga et Ledoux, 1978; Rensink *et al.*, 1977; Weiskrantz, 1986], relevées par Brooks [Brooks *et al.*, 1998], démontrent que l'être humain ferait appel à un ensemble de représentations distribuées plutôt qu'à un seul modèle monolithique, ce qui donnerait plus de support à l'approche comportementale qu'à l'approche délibérative qui est monolithique.

2.2.2 Perception multimodale et orientée selon l'action

Le principe de base de la perception orientée selon l'action est d'extraire de l'environnement seulement l'information nécessaire à l'exécution d'une tâche en particulier [Arkin, 1998]. La nature de la perception dépend donc de la tâche en cours d'exécution. Si un robot autonome doit exécuter plusieurs tâches potentiellement différentes, il apparaît clair que son architecture de contrôle devra supporter plusieurs types de perception, d'où l'intérêt d'une perception multimodale implicant des capteurs extéroceptifs comme la vision ou l'audition, mais aussi capteurs proprioceptifs comme la position des actionneurs qui génèrent les mouvements du robot. Une perception multimodale et orientée selon l'action apparaît donc comme un élément nécessaire dans le développement d'une architecture de contrôle capable d'orienter son attention.

Les schémas perceptuels peuvent être appliqués pour décrire une perception modulaire et multimodale. Selon Arkin [1998], un schéma perceptuel doit être créé de façon à produire l'information nécessaire à une tâche et est nécessairement lié à au moins un schéma moteur. Il relève également plusieurs points intéressants pour la perception modulaire de la théorie des schémas d'Arbib [1992] :

- Les schémas sont un ensemble de multiples processus actifs et concurrents axés sur des activités perceptuelles différentes.
- Les schémas comportent autant le contrôle que la connaissance requise pour compléter une tâche perceptuelle.
- Les schémas forment un réseau actif de processus fonctionnant de façon distribuée spécifique à la situation et à l'ensemble des intentions d'un agent.
- La théorie des schémas fournit la base pour des languages pouvant définir une perception orientée selon l'action.
- Le niveau d'activation associé à un schéma peut être lié au degré de croyance dans un événement perceptuel particulier.

La notion de déclencheur perceptuel (*perceptual trigger* dans [Arkin, 1998]) est intéressante pour la configuration dynamique d'une perception multimodale. En effet, un déclencheur perceptuel est un événement qui modifie l'état comportemental ou perceptuel du robot. Par exemple, le fait de détecter que le robot évolue dans le noir, ce qui rend l'utilisation de la vision inutile, pourrait désactiver le sous-système visuel. Cette notion propre aux schémas perceptuels est évidemment liable aux éléments notés à la section 2.1.2.

Une perception multimodale implique non seulement l'utilisation de capteurs de natures différentes, mais également l'analyse et la production de différentes représentations à partir d'un même capteur. Concrètement, la voiture autonome *Junior* [Montemerlo *et al.*, 2008], qui a remporté la deuxième position au DARPA Urban Challenge, implémente une forme de perception multimodale pour ses capteurs lasers. Pour éviter les obstacles et accumuler une carte de son environnement, le système utilise plusieurs techniques différentes selon la tâche à effectuer. Pour la plupart des obstacles, le système surveille les lectures d'un seul LIDAR Velodyne pour des variations à la verticale dans le champ tridimensionnel observé par une technique très dépendante de la nature du capteur. Or, puisque celui-ci possède des angles morts, il utilise également une batterie de capteurs lasers à plus courte portée pour détecter les obstacles à proximité. De plus, pour la détection d'obstacles dynamiques,

le système compose en un seul balayage plusieurs lectures laser de façon à simuler un seul balayage qui aurait été fait par un capteur à deux dimensions sur 360 degrées. Ainsi, un seul algorithme peut être utilisé et testé peu importe la nature des capteurs. Donc, même si une carte globale assemblée à partir de plusieurs cartes locales est accumulée tout au long du parcours du véhicule, ce n'est pas le seul mode de représentation d'obstacles qui est utilisé à toutes les tâches. Les résultats obtenus par l'équipe montrent que cette approche est viable. En effet, aucun obstacle n'a été rencontré, incluant des bordures par dessus lesquelles le véhicule aurait pu passer sans réel danger.

Tel que mentionné à la section 2.1.2, Wyatt et Hawes [2008] ont soulevé quelques problèmes ou défis dans l'utilisation d'une perception multimodale, notamment en ce qui concerne la liaison de différents stimuli de natures différentes et particulièrement dans un système distribué. Le CAS met en œuvre une mémoire à long terme pour lier des stimuli provenant de sous-architectures indépendantes. Les sous-architectures regroupent généralement des modules qui ont besoin d'échanger de l'information régulièrement, mais des hiérarchies de types permettent de lier par exemple le mot «bleu» reconnu par un système de parole à la couleur bleue perçue par le système de vision. Cet aspect de leur système est détaillé dans d'autres travaux [Jacobsson *et al.*, 2008; Vrečko *et al.*, 2009].

2.2.3 Intégration de motivations

Kismet

Tel que mentionné dans la section 2.1.2, *Kismet* possède un système de motivation pour moduler l'attention et l'activation de comportements. Cet aspect du système est brièvement décrit dans [Breazeal et Scassellati, 1999], mais est plus détaillé dans [Breazeal et Scassellati, 2000]. Leur approche, inspirée des observations du comportement animal de Lorenz [1973], est composée de deux sous-systèmes : les pulsions (*drives*) et les états expressifs et émotionels. Le premier représente l'objectif du robot tandis que le deuxième le degré de satisfaction dans l'atteinte du premier. Les pulsions influencent donc l'expression du robot en plus de réguler l'activation des comportements.

CAST

Avec CAST, les motivations du robot sont produites par des sous-architectures de motivations. Ces sous-architectures font généralement une planification à l'interne des tâches de traitement et d'action à accomplir. Dans le cas de PlayMate/Explorer [Hawes *et al.*, 2009c], présentée à la figure 2.3, la motivation provient d'une sous-architecture de planification appelée *planning SA*, qui coordonne les actions d'autres sous-architectures pour la

navigation autonome (*nav SA*) ou la production de dialogue (*comsys SA*). La source d'une motivation est quant à elle généralement produite par une sous-architecture d'association (*binding SA*). Dans [Hawes *et al.*, 2009a], les auteurs donnent l'exemple d'une association possible entre comprendre une commande d'un utilisateur comme "Prends ce livre" provenant d'une sous-architecture de reconnaissance de la parole, et la détection d'un livre dans le champ de vision du robot provenant d'une autre sous-architecture. Cette association peut être récupérée par une sous-architecture de motivation, qui elle produira un but précis de navigation et/ou de manipulation pour une ou plusieurs sous-architectures. Dans ces exemples d'utilisation de CAST, la sous-architecture de motivation a donc à la fois un rôle de planification haut-niveau et un rôle de coordination de modules plus bas-niveau. Par contre, CAST en tant que tel n'empêche pas la division de la production de buts et de la coordination de ceux-ci en deux sous-architectures, le choix revenant à l'utilisateur de l'infrastructure.

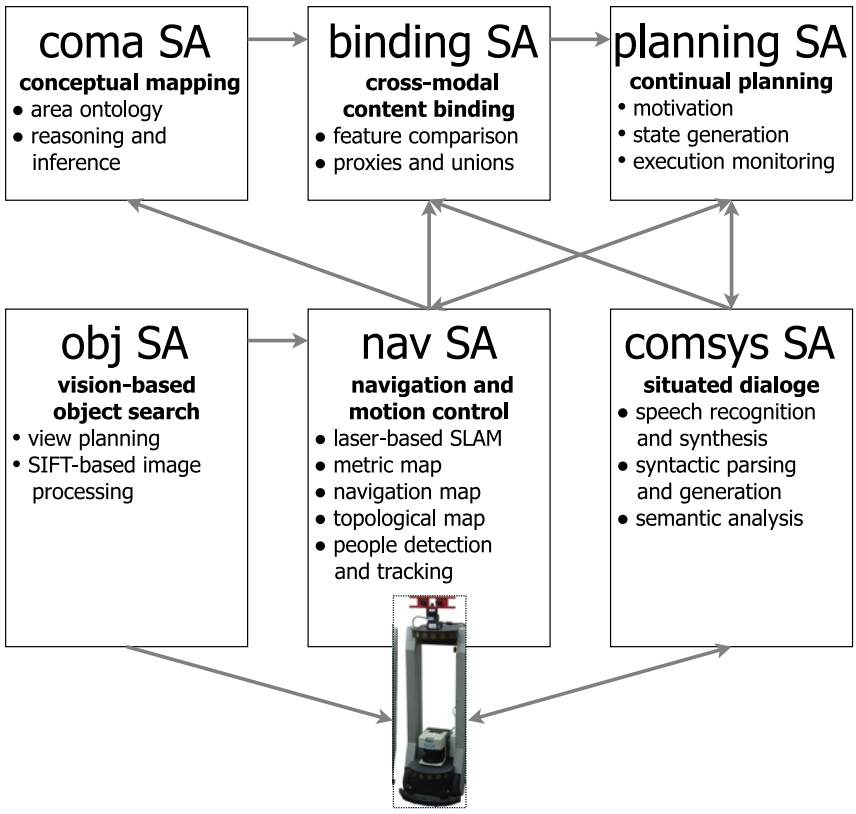


Figure 2.3 L'architecture d'Explorer, un exemple d'utilisation de CAST [Hawes *et al.*, 2009c].

DIARC [Scheutz *et al.*, 2007], pour *Distributed Integrated Affect, Reflection and Cognition architecture*, est une architecture de contrôle qui intègre plusieurs caractéristiques permettant des interactions naturelles entre humains et robots. Une vue d'ensemble est disponible à la figure 2.4. Son implémentation utilise *Agent Development Environment* (ADE) [Andronache et Scheutz, 2006; Scheutz, 2006], une infrastructure de développement utilisant Java et permettant le développement d'applications robotiques distribuées autant virtuels que réels. DIARC est une architecture robotique affective, ce qui implique qu'autant la reconnaissance que la production d'affect [1] sont présentes pour obtenir des interactions plus harmonieuses entre humains et machines. Les affects peuvent être exprimés et reconnus autant par des messages verbaux que non-verbaux (gestes, changement de tons dans la voix, ...). Ils peuvent également être utilisés par la partie délibérative de l'architecture pour moduler la partie non-délibérative par le biais de mécanismes motivationnels, en modifiant des buts ou en altérant les préférences de sélection d'actions, par exemple. Dans DIARC, les affects peuvent donc être utilisés pour modifier l'état des motivations non seulement par les lectures des capteurs mais aussi par l'état d'autres sous-systèmes. Au centre de l'architecture se trouve l'*Affective Action Interpreter* [Scheutz *et al.*, 2006], un environnement d'exécution de scripts permettant de décrire des tâches haut niveau par des séquences d'actions plus ou moins primitives. Dans leur infrastructure, cet interpréteur est présent dans trois modules, les *Goal*, *Task* et *Action Managers*. Les scripts peuvent s'exécuter de façon concurrente et réagir à différents événements provenant autant du système perceptuel ou des résultats d'autres scripts en cours d'exécution. L'échec d'un script, p. ex., lorsqu'un but devient inatteignable, entraîne une modification de l'état affectif du robot. La modification des affects n'influence pas directectement les actions ou l'exécution d'un script, mais entraîne plutôt la réorganisation des priorités des objectifs en cours. Comme chaque objectif a un script qui lui est associé et que plusieurs scripts peuvent être exécutés simultanément, plusieurs objectifs peuvent être atteints à la fois. Or, si deux scripts nécessitent les même ressources moteurs, p. ex., activer les manipulateurs, celui ayant la plus haute priorité est exécuté exclusivement. L'approche est donc analogue à une arbitration par priorité dans une architecture comportementale multiniveau. Pour gérer l'accès à ces ressources, un mécanisme classique d'obtention d'un verrou est utilisé. Dans le cas où plusieurs ressources seraient nécessaires, il est donc important de suivre un ordre précis d'obtention de verrous pour éviter les interblocages, exactement comme en programmation concurrente. Selon les développeurs de l'interpréteur, il existe des mé-

[1] États émotifs, émotions, sentiments.

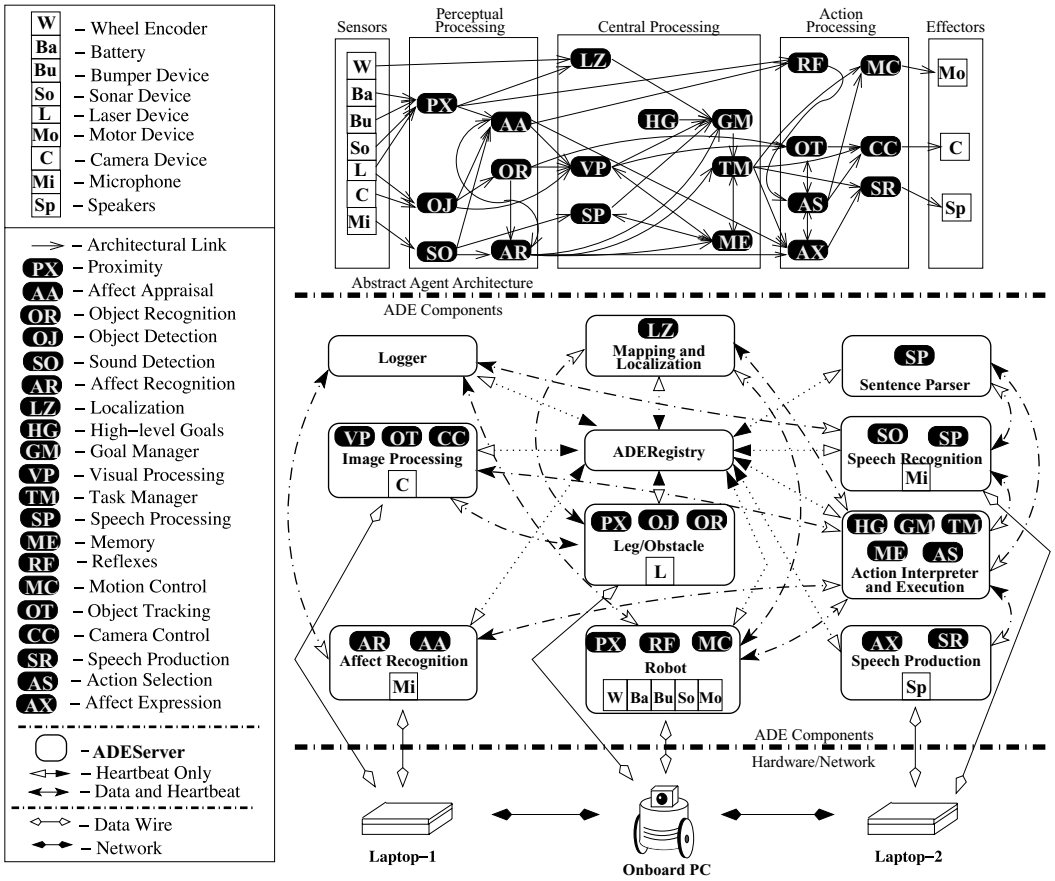


Figure 2.4 L'architecture DIARC et son infrastructure [Scheutz *et al.*, 2007].

canismes pour surveiller l'état du moteur d'exécution et le réparer en cas de défaillance [2].

MBA

Spartacus [Michaud *et al.*, 2007a] est également un robot autonome, cette fois-ci mobile, intégrant un sous-système de motivations. MBA, pour *Motivational Behavioral Architecture*, est une architecture de contrôle comportementale multiniveau où les motivations viennent balancer efficacement l'adaptation du robot aux contingences de l'environnement et l'accomplissement de ses objectifs. Un schéma de la structure de MBA est disponible à la figure 2.5. Les sources motivationnelles sont divisées en trois catégories : instinctives, rationnelles et émotionnelles. Les sources instinctives représentent des opérations de base pour le robot, comme la surveillance du niveau des piles. Les sources rationnelles influencent plutôt des tâches cognitives comme la planification. Finalement, les sources émotionnelles surveillent les situations transitionnelles ou conflictuelles entre différentes tâches. Le sys-

[2]Explication issue d'une conversation privée avec Paul Schermerhorn.

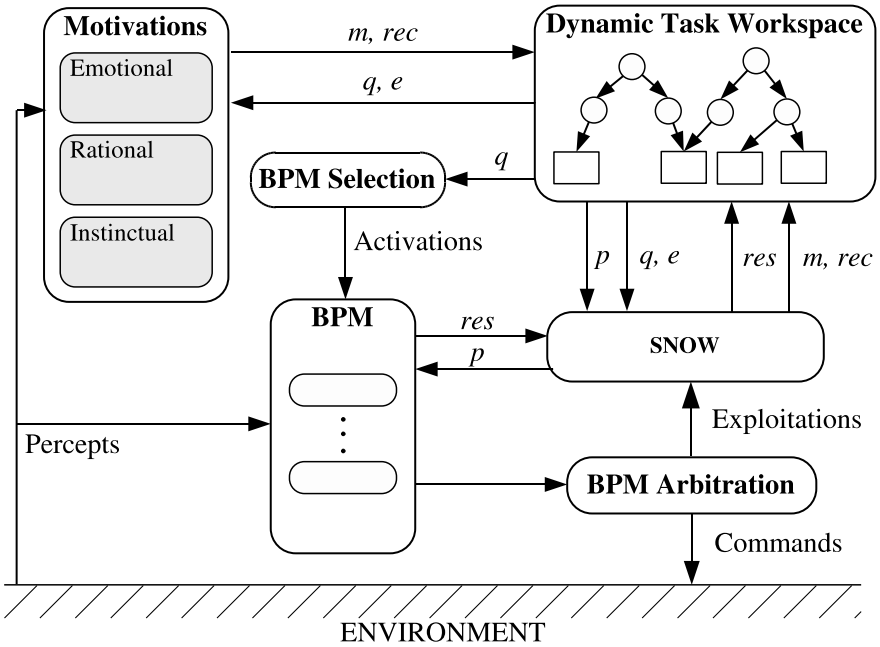


Figure 2.5 L'architecture MBA [Michaud *et al.*, 2007a]. Les cases de la section **BPM** représentent des *Behavior-Producing Modules*, ou comportements, indépendant. Le diagramme du *Dynamic Task Workspace* est une représentation d'un ensemble de tâches et sous-tâches à accomplir. La variable m représente une requête de modification, rec une recommandation, q une interrogation par rapport à une tâche, e un événement, p un paramètre de tâche et res un résultat suite à l'exploitation d'un comportement.

tème motivationnel travaille de paire avec le *Dynamic Task Workspace* (DTW), qui gère différentes tâches de haut et bas niveaux selon leurs interdépendences. Le DTW, après réception de différentes requêtes du système motivationnel, émet des recommandations auprès du système de sélection des *Behavior-Producing Modules* (BPMs) qui, après arbitration, peuvent envoyer des commandes motrices. Le *System Know-How* (SNOW) est un adaptateur entre les BPMs et le DTW qui permet de découpler les représentations propres à ces modules de façon à ce qu'ils puissent fonctionner de façon indépendante. Le SNOW est surtout un élément de l'architecture logicielle du système et ne participe pas vraiment à la structure décisionnelle. Cette architecture est une généralisation de EMIB [Michaud, 2002], pour *Emotion and Motivation for Intentional selection and configuration of Behaviour-producing modules*, qui dans son implémentation ne possèdait pas le SNOW.

Les motivations peuvent être utilisées pour surveiller l'état d'avancement du robot de façon similaire au problème décrit par Scheutz et Andronache [2004] à la section 2.1.2. Avec EMIB, une motivation de détresse (*Distress*) surveille l'exploitation des comportements

pour détecter des séquences pouvant indiquer l'apparition d'un problème. Par exemple, dans l'environnement de simulation *BugWorld* [Almàssy, 1993], l'exploitation simultanée des comportements *Emergency* et *Avoid* pendant de trop longues périodes de temps faisait augmenter le niveau de *Distress* lorsque le robot se retrouvait coincé. À partir d'un certain seuil, ce module motivationnel désactivait un des deux comportements pour tenter une nouvelle stratégie.

HBBA

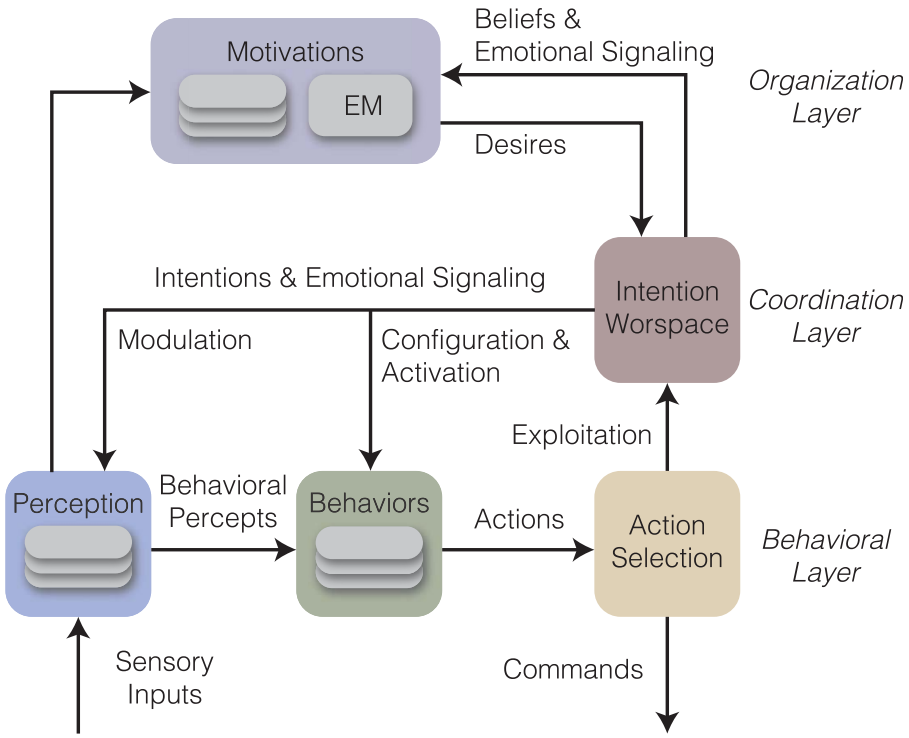


Figure 2.6 L'architecture HBBA, tirée de [Ferland *et al.*, 2014].

HBBA, pour *Hybrid Behavior-Based Architecture* [Michaud *et al.*, 2010b], est une évolution directe de MBA. Elle est illustrée à la figure 2.6. Comme pour MBA, des modules motivationnels haut-niveau produisent des désirs décrivant des tâches à accomplir (par exemple, se rendre à un endroit en particulier) ou de l'information à acquérir (par exemple, la détection de visages ou la localisation de sources sonores). Les désirs sont ajoutés à l'*Intention Workspace* (IW), qui permet de décrire à tout moment les désirs provenants des modules motivationnels. Un sous-module, appelé l'*Intention Translator*, se charge de convertir la description de ces désirs en intentions concrètes pour le robot à partir d'une base de données de stratégies décrivant les capacités du robot. Plusieurs désirs concurrents peuvent co-exister à l'intérieur de l'IW, mais un seul désir conflictuel peut être activé à la fois. La sélection se fait en fonction de l'intensité des désirs, qui décrit l'importance qu'attache un

module motivationnel à ce que ce désir soit comblé. Au lieu d'avoir des sources motivationnelles distinctes pour les émotions, un module séparé de génération d'émotions inspiré des travaux de Raïevsky et Michaud [2008] est présent, et peut venir influencer l'état des désirs dans l'IW. L'IW a également pour tâche de surveiller l'exploitation des comportements et de faire correspondre celle-ci aux désirs actifs, de façon à pouvoir communiquer aux modules motivationnels si leurs désirs ont été considérés dans les intentions du robot, et si ces intentions se sont traduites en exploitation concrète d'un ou plusieurs comportements. Ce sont les disparités entre désirs actifs et désirs exploités qui viennent informer le module de génération d'émotions, qui lui ensuite peut moduler l'intensité des désirs ignorés, de façon à provoquer un changement dans les intentions du robot. HBBA inclut également un module de mémoire épisodique (EM dans la figure 2.6) [Leconte *et al.*, 2014] qui permet d'emmagasiner à court et à long terme autant l'information produite par les modules perceptuels que les changements observés dans l'activation et l'exploitation des désirs. Le rappel de ces événements peut ensuite être utilisé par les modules motivationnels pour adapter leur production de désirs, par exemple en répétant les recommandations d'actions qui ont contribué à la réalisation d'une tâche dans des épisodes passés.

Comme le décrit cette section, il existe déjà plusieurs approches différentes pour gérer un grand nombre de processus perceptuels et décisionnels à l'aide de la sélection d'attention dans une architecture de contrôle. Or, elles se limitent toutes à une sélection des modules à exécuter et ne permettent donc pas de réduire partiellement leur charge de calcul. Les chapitres suivants de cette thèse présente une nouvelle approche permettant ce type de modulation.

CHAPITRE 3

Natural Interaction Design of a Humanoid Robot

Avant-propos

Auteurs et affiliation :

F. Ferland : étudiant au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a été responsable de l'intégration logicielle complète du robot, ainsi que du déroulement des expériences présentées dans cet article.

D. Létourneau : professionnel de recherche, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a contribué à la conception électrique et l'assemblage d'IRL-1 et AZIMUT-3.

A. Aumont : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a contribué au contrôle par impédance et à la compensation de gravité des bras d'IRL-1.

J. Frémy : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a développé la première version de l'algorithme de perception des forces par le chassis d'AZIMUT-3.

M.-A. Legault : professionnel de recherche, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il était responsable de la conception mécanique d'IRL-1 et AZIMUT-3.

Michel Lauria : professeur, Haute école spécialisée de Suisse occidentale (HES-SO). Il a supervisé les travaux de Julien Frémy en plus de la conception mécanique d'IRL-1 et AZIMUT-3.

François Michaud : professeur, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a contribué à la conception de l'architecture HBBA et à la révision de cet article.

Date d'acceptation : 23 juillet 2012

État : version finale publiée (28 janvier 2013)

Revue : Journal of Human-Robot Interaction

Référence : [Ferland *et al.*, 2012]

Titre français : Conception d'interaction naturelle d'un robot humanoïde

Contribution au document : Cet article contribue à la thèse en décrivant la conception physique de la plateforme robotique utilisée ainsi que trois études de faisabilité d'interaction humain-robot, dont la dernière concernant l'impact sur le délai de réaction du robot lorsqu'un filtrage perceptif est effectué.

Résumé en français : La conception de robots interagissant naturellement avec des personnes nécessite l'intégration de technologies et d'algorithmes pour de multiples modalités de communication comme la production de gestes, le mouvement, les expressions faciales et les interfaces graphiques. Pour comprendre les interdépendances de ces modalités, l'évaluation d'une conception intégrée par le biais d'études de faisabilité procure de l'information à propos des considérations clées à apporter au robot et à ses scénarios d'interaction potentiels, permettant ainsi de raffiner la conception de manière itérative avant que des études de plus grande envergure puissent être effectuées. Cet article présente trois études de faisabilité avec IRL-1, un nouveau robot humanoïde intégrant des actionneurs compliants pour le mouvement et la manipulation, ainsi que des capacités de vision et d'audition artificielles et des expressions faciales. Ces études explorent des capacités distinctes d'IRL-1, incluant la possibilité d'être guidé physiquement en percevant les forces externes par le biais d'actionneurs élastiques utilisés dans la direction de sa plateforme omnidirectionnelle ; l'intégration de la vision, du mouvement et de l'audition dans une interface de téléprésence augmentée ; et l'influence des délais de réaction à des sons dans l'environnement. En plus de démontrer comment ces capacités peuvent être exploitées en interaction humain-robot, cet article illustre les interrelations intrinsèques entre la conception et l'évaluation d'IRL-1, comme l'influence du point de contact lors du guidage physique de la plateforme, la synchronisation entre les données sensorielles du robot et sa représentation graphique, ou la perception selon sa vitesse de réponse par les mouvements du visage lorsque des processus intensifs d'un point de vue informatique sont utilisés. L'article décrit également quels genres d'expériences plus avancées pourront être conduites avec la plateforme.

Une vidéo montrant le déroulement des trois expériences avait également été jointe avec la publication. Celle-ci fait environ 80 Mo et est disponible publiquement sur le site internet du journal à l'adresse suivante :

`http://hri-journal.org/index.php/HRI/article/view/65/48`

Abstract

Designing robots that interact naturally with people requires the integration of technologies and algorithms for communication modalities such as gestures, movement, facial expressions and user interfaces. To understand interdependence among these modalities, evaluating the integrated design in feasibility studies provides insights about key considerations regarding the robot and potential interaction scenarios, allowing the design to be iteratively refined before larger-scale experiments are planned and conducted. This paper presents three feasibility studies with IRL-1, a new humanoid robot integrating compliant actuators for motion and manipulation along with artificial audition, vision, and facial expressions. These studies explore distinctive capabilities of IRL-1, including the ability to be physically guided by perceiving forces through elastic actuators used for active steering of the omnidirectional platform ; the integration of vision, motion and audition for an augmented telepresence interface ; and the influence of delays in responding to sounds. In addition to demonstrating how these capabilities can be exploited in human-robot interaction, this paper illustrates intrinsic interrelations between the design and evaluation of IRL-1, such as the influence of the contact point in physically guiding the platform, the synchronization between sensory and robot representations in the graphical display, and facial gestures for responsiveness when computationally expensive processes are used. It also outlines ideas regarding more advanced experiments that could be conducted with the platform.

3.1 Introduction

Valli [2008] defines natural interaction "in terms of experience : people naturally communicate through gestures, expressions, movements, and discover the world by looking around and manipulating physical stuff ; the key assumption here is that they should be allowed to interact with technology as they are used to interact with the real world in everyday life, as evolution and education taught them to do." *Natural* is thus synonymous with *normal interactions in everyday life.*

For the last 15 years, our research laboratory has been interested in the design of natural interactive mobile robots. These platforms evolved from small robots, such as an autonomous rolling robot used in child development studies [Michaud *et al.*, 2005a, 2007b; Salter *et al.*, 2010] and a teleoperated interactive platform used as an imitation agent with children with low-functioning autism [Duquette *et al.*, 2008], to a telepresence robot for home care assistance [Michaud *et al.*, 2010a] and a robot that can interact autonomously

with people in open settings [Michaud *et al.*, 2007a]. In these studies, the experimental settings were conducive to natural interaction : toddlers discovered how to interact with a rolling robot ; rehabilitation professionals evaluated whether they could conduct clinical evaluations through a telepresence robot ; an interactive autonomous robot got to face the challenges of going to a conference as a regular attendee. These studies allowed us to develop and evaluate the robots holistically, that is, by considering the robot at a human-robot-context systems level [Lee *et al.*, 2009; Nelson et Stolterman, 2003] in order to identify technological specifications and requirements that can affect human-robot interaction (HRI). This led to the development of innovative technologies and algorithms for natural interaction, such as the following :

Differential Elastic Actuators (DEAs) for safe and compliant actuation [Lauria *et al.*, 2008]. A DEA acts as an active elastic element that can inherently absorb shocks, perceive the forces from the environment on the robot, and control the forces applied in response. This capability, also known as interaction control [Buerger et Hogan, 2005], is essential for a robot that interacts naturally with people in open settings, because physical contact with the robot must be sensed and processed appropriately. DEAs are conceptually similar to Series Elastic Actuators (SEAs) [Robinson, 2000; Williamson, 1995], which are used on the Meka M1[1] , Cody [Jain et Kemp, 2009], and Nexi-MDS[2], but use a differential coupling (harmonic drive) instead of a serial coupling of a high impedance mechanical speed source (an electrical DC brushless motor) and a low impedance mechanical spring (a passive torsion spring). A non-turning sensor connected in series with the spring measures the torque output of the actuator. This results in a more compact and simpler solution for controlling mechanical elasticity and viscosity in accordance with an admittance control scheme [Buerger et Hogan, 2005; Hogan, 1985], with performance similar to that of SEAs.

ManyEars, a sound source localization, tracking, and separation system [Valin *et al.*, 2007a,b] that uses an array of eight microphones. People interacting with a robot with voice generation capabilities naturally talk back to the robot, as observed, for instance, in Michaud, Laplante, et al. 2005a. ManyEars provides an enhanced speaker signal for improved speech

[1]The Meka M1 was was produced by Meka Robotics, a company bought by Google in 2013

[2]Nexi-MDS was produced at the MIT Media Lab Personal Robots Group in collaboration with Xitome Design and University of Massachusetts Amherst.

and sound recognition in real-world settings. It can simultaneously localize and track up to four sound sources and separate three, all in reverberant and noisy conditions. Only a very limited set of HRI platforms, such as the Nexi-MDS, which has a four-microphone array and a wearable microphone, and the Snackbot [Lee *et al.*, 2009], which uses the Acoustic Magic microphone array, are equipped - with limited capabilities - with artificial audition systems for interacting with people. ManyEars is also the framework on which the HARK open source software [Nakadai *et al.*, 2008] is built.

As observed during the holistic design of Snackbot [Lee *et al.*, 2009], the design of an interactive robot is extremely difficult, even when conventional technologies are used. The main challenge is that there is as much to learn from the interaction between the integrated technologies as from the embodied human-robot interaction, in addition to interdependent effects. In the field of human-computer interaction, "interaction design" is the practice of designing digital products, environments, systems, and services for human-computer interaction [Cooper *et al.*, 2007]. Because additional factors must be taken into consideration in embodied interaction, the question then becomes how "interaction design" should be done in the context of natural HRI.

This paper presents one avenue that we are exploring with the design of IRL-1 [Michaud *et al.*, 2010b], a humanoid platform with an expressive face, an orientable head, an eight-microphone array, two arms with 4 degrees of freedom (DOF) each, and grippers, which are all mounted on an omnidirectional, non-holonomic mobile platform. DEAs are used for interactive control of the 4-DOF arms and active steering of the mobile platform, and ManyEars is used to provide artificial audition capabilities. The technologies designed to enable natural interaction were developed independently, and thus may not function seamlessly when integrated onto the same platform. By conducting feasibility studies (also referred to as case studies in Tsui and Yanco 2009) with the robot in different interaction scenarios, we studied the capabilities of the platform to facilitate this integration following an incremental and iterative design process, along with evaluating ideas of interaction scenarios before conducting in-depth HRI studies. These feasibility studies allowed us to identify, investigate, and validate the influences of IRL-1's specific capabilities (such as the ones provided by DEAs and ManyEars) on the dynamics of interacting with the platform. After presenting IRL-1 and its natural interaction capabilities, this paper describes feasibility studies examining the ability to physically guide the platform, the integration of directed vision and audition for telepresence, and the influence of integrating software

processes with a selective attention mechanism under the constraint of limited computing resources on the robot's responsiveness in HRI. In addition to demonstrating the capabilities of the platform, observations and insights about design and evaluation of their current and future integration work are provided.

3.2 IRL-1

Figure 3.1 shows IRL-1, an omnidirectional platform with two compliant arms and a humanoid torso. Each of these elements provides important capabilities in terms of natural HRI.

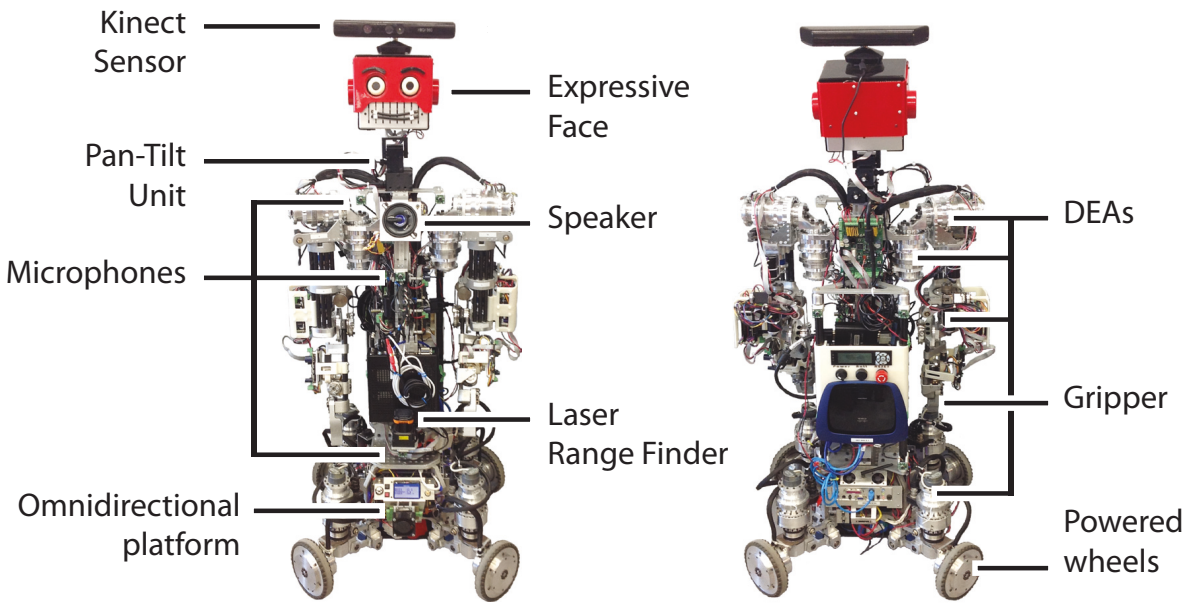


Figure 3.1 Photographs of the front and back of IRL-1. The robot has 30 DOF and 55 sensors.

Omnidirectionality is the ability to move in all directions without changing orientation, which is useful when moving in tight spaces or crowded areas. Instead of using Mecanum wheels as does the Cody robot [Jain et Kemp, 2009], or powered caster wheels as do PR2 [Wyrobek *et al.*, 2008] and Rollin' Justin [Borst *et al.*, 2009], omnidirectionality on IRL-1 is provided by DEA-steered ($\pm 90°$) powered wheels with a lateral offset from their attachment points. They are lighter, more precise and mechanically simpler than Mecanum wheels, which involve multiple rollers oriented 45° from the rotation axis of the wheel and can slide laterally. They also provide a built-in horizontal suspension system that is absent when powered caster wheels are used. This also makes physical interaction possible from almost any point on the platform, as long as the force applied is not parallel to all of

the wheels' propulsion axes. By having off-centered steerable wheels, a force applied at a wheel's contact point can be measured as torque on its steering axis. By combining these torque measurements into a global model of forces applied to the chassis, the platform can react to an external force by moving in its direction [Frémy *et al.*, 2010]. The use of off-centered wheels also lowers the height of the chassis, and a passive vertical suspension made of four Rosta springs is used to connect the steerable wheels to the platform's chassis, thus helping to keep the wheels in contact with the ground on uneven surfaces. The platform has a 34 kg payload and can reach a maximum velocity of 1.47 m/s.

DEAs are also used to provide compliance to IRL-1's arms. Each arm is attached to the torso and has 4 DOF (three in the shoulder and one in the elbow). Impedance control of each joint enables an infinite combination of arm behaviors, from zero impedance for free movement with gravity compensation to high stiffness constraining the arms to precise positions or ranges of movement. The arms can sustain impacts with humans or objects, and they can be controlled from low to high admittance with gravity compensation [Legault *et al.*, 2008]. A gripping tool serves as a hand at the end of each arm. IRL-1's arms provide sufficient capabilities for HRI studies, even though they have a smaller number of DOF than other compliant mobile manipulation platforms, such as Meka B1, Cody [Jain et Kemp, 2009], Nexi-MDS, Rollin' Justin [Hirzinger *et al.*, 2004; Ott *et al.*, 2006], and PR2 [Wyrobek *et al.*, 2008].

IRL-1's torso is equipped with one loudspeaker, a Hokuyo UTM-30LX laser range finder (30 m range, 270° angle and 25 ms/scan, 0.4 m from the ground), one Kinect motion sensor (providing a 3D point cloud with color for a 640×480 image with an angular field of view of 57° horizontally and 43° vertically, up to 5 m and at 30 frames per second) and a robot head installed on a pan-tilt unit (FLIR PTU-D46-17, actuated using step motors, pan $\pm 128°$, tilt $[-17, +25]°$). Figure 3.2 illustrates facial expressions that the robot can make using nine servo-motors (four for the mouth, three for the eyes, and two for the eyebrows). These are not as advanced as the facial expression capabilities of platforms like Meka B1 and Nexi-MDS, but they are sufficient to convey the intended emotion. IRL-1's robot head has been used in other HRI studies [Carter *et al.*, 2009; Rich *et al.*, 2010; Shayganfar *et al.*, 2012].

IRL-1 integrates a broad set of features that could be beneficial for HRI studies, with 30 DOF (8 DOF on the mobile base, 10 DOF for the arms and grippers, 11 DOF for the head and the neck, and the loudspeaker) and 55 sensors (position encoders for all 29 actuated DOF plus 16 torque sensors for the ones actuated using DEA, the laser range finder, the Kinect sensor and the eight microphone array). The robot is equipped with two Mini-ITX

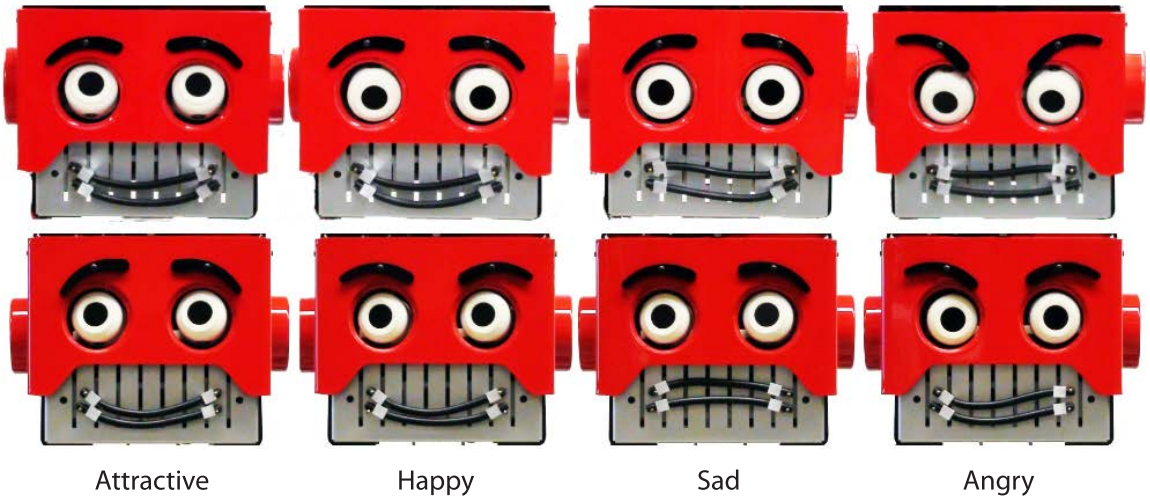


Figure 3.2 Photographs of IRL-1's facial expressions : attractive, happy, sad, and angry. Each column shows two variations of a single emotion. The robot's facial expressions are programmed to switch between variations of a single emotion every five seconds.

computers, both running Linux. The first one, a 2.0 GHz Core 2 duo processor with 2 GB of RAM, is located in the mobile base and hosts processes related to the robot's base, such as motion control, simultaneous localization and mapping (SLAM), and path planning. The second one, a 2.67 GHz Core i7 quad core processor with 4GB of RAM, is located in front of the torso. It hosts the arm controllers, head expression and orientation, sensor systems such as the base laser range finder, Kinect camera and microphone array, and high-level cognitive processes. The remaining hardware architecture includes 20 distributed controller modules for sensing, power, and low-level control, communicating with each other through a 1 Mbps CAN bus [Michaud *et al.*, 2005b]. Nickel-metal hybrid batteries provide power to the mobile base for around half an hour of autonomy at maximum speed or for about 2 hours when the robot is immobile. The torso is powered by a separate pair of 10 Ah $LiFePO_4$ battery packs, providing up to 10 hours of autonomy when the robot's arms are stationary.

3.3 Physical Interaction Feasibility Study

Physical interaction with a small robot can be relatively simple, intuitive, and natural, as has been observed with children and toddlers [Michaud *et al.*, 2005a, 2007b]. But for large and heavy robots such as IRL-1 (1.4 m high and around 70 kg), an active method is required for the robot to react to human pushing or pulling. Remote controls are commonly

used to operate such robots. However, leading the robot directly by the hand or arm has been shown to be more desirable in many ways [Chen et Kemp, 2011, 2010]. In that regard, the contact points for direct physical interaction (DPI) should not be limited to specific parts of the robot. In previous work, we demonstrated that this capability can be achieved with our omnidirectional base [Frémy *et al.*, 2010], and here we investigate whether it is possible with IRL-1's upper torso installed on the base.

Figure 3.3 illustrates the experimental setup. The objective is to make the robot rotate 180° by going through a three-point turn trajectory under two conditions :

- Physical interaction : Participants can push or pull the robot directly, and the applied forces sensed by DEAs are used for active steering of the omnidirectional base. IRL-1's arms are slightly extended to keep its grippers at 0.45 m from the shoulders. Torque measurements from DEAs on the arms are not exploited to detect physical interaction, but pushing or pulling the arms or the grippers can generate enough force on a wheel's contact point to be detected by the DEA-steered powered wheels.

- Remote control : The left analog control stick commands both linear velocity in X and angular velocity around Z. Participants were free to move around to get a better view of the robot's movement.

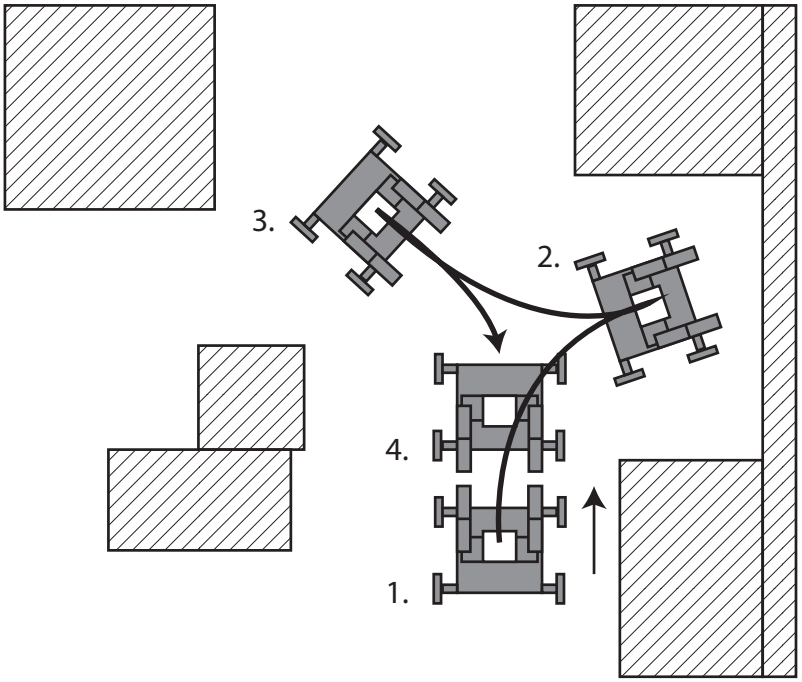


Figure 3.3 Physical interaction experiment involving a three-point turn trajectory.

The experiments took place in a cluttered office space of 10 m^2. Instructions for the robot and the remote control were given to the participants, and they had up to 15 minutes to practice with the control conditions before performing the experiment. A convenience sample of six male engineers (aged from 22 to 26 years) were permitted five attempts with each control method to perform the maneuver while the task completion time and the number of collisions were measured.

Participants performed faster with the remote control ($\bar{x} = 23.93$ s, $\sigma = 10.98$ s) compared to physical interaction ($\bar{x} = 34.67$ s, $\sigma = 14.99$ s). This can be explained by the fact that the maximum velocity can easily be reached using the remote control. Also, participants were more careful not to damage the robot when interacting physically with it, and more collisions occurred when participants used the remote control (three versus none). This last observation concurs with Chen and Kemp [2011; 2010], that is, collisions occurred significantly less often when participants performed their tasks with a DPI instead of a gamepad remote control.

Comments from the participants revealed that while it was easy to move the robot on a straight path, it was difficult to send the mobile base on a curved path or to switch from a left-handed curve to a right-handed one. Torque measurements from the steering axes' actuators can be noisy and often contain a constant bias difficult to predict and mostly caused by the weight of the robot's torso itself. While we could dynamically estimate the center of gravity of the robot based on all of its joint positions, unmonitored suspension travel would render this estimate almost useless for predicting the actual effect of the weight of the robot's upper body on the force-guiding algorithm. A minimal torque threshold of 0.800 Nm was set to compensate for this bias : While this effectively masks noise and bias from the torque measurements, it explains why more force must be applied to move the platform. This is especially apparent when rotating the platform by physically interacting with its arms : the longest moment arm to apply torque around the robot's Z axis passes through the shoulders, which even when actuated with high impedance absorb part of the torque. Suspension travel and force absorption by the elasticity of the arms can be avoided by applying external forces directly to the mobile platform's steering actuators, but these contact points are lower and not as easily accessible. Therefore, for a platform where elasticity and force-controlled actuation are present both at the locomotion and arm levels, the contact point for physical interaction will influence performance. To improve sensitivity to physical interaction with the arms, we can use position and torque measurements directly from the arms' DEAs to examine how these can be used together with the torque measurements from the base to allow smooth, precise, and responsive physical in-

teraction with the robot. Since there is an interaction between the two compliant systems (the omnidirectional base and the arms), it will also be important to measure the contact point with people who are naturally interacting with the platform. Upgrading the suspension of the omnidirectional base and using compliant actuators that do not exploit springs as the elastic elements [Fauteux *et al.*, 2010] would also help. In spite of these limitations, two participants mentioned that they found pushing the robot more intuitive than using the remote control, and no comments were made to the contrary. One of these participants noted that the remote control scheme was very similar to common car-racing videogames (which made it easy for him to understand), but the learning curve still appeared to be easier with physical interaction.

3.4 Augmented Teleoperation Feasibility Study

Natural interaction in robot telepresence can be addressed from two perspectives : from the perspective of the person interacting with the robot at the remote location or from the perspective of the distant operator. Using humanoid robots such as IRL-1 for telepresence applications provides capabilities such as motion, gesture, force sensing, and sound source localization that go beyond what is available on more conventional telepresence platforms, such as Rovio, Vgo, TiLR, Giraffe, RO-7i, Sparky Jr., Texai and QB [Guizzo, 2010], or our telepresence robot for home care assistance [Michaud *et al.*, 2010a]. For the distant operator, the main challenge of telepresence is understanding an environment in which one is not physically present [Nielsen *et al.*, 2007]. Situation awareness [Endsley, 1988; Labonté *et al.*, 2010; Scholtz *et al.*, 2004, 2005] and cognitive load [Labonté *et al.*, 2010; Nielsen *et al.*, 2007] are primary concerns, especially with a humanoid robot with many sensors and DOF. Assuming that the remote operation station is a computer, the challenge then becomes how these capabilities can be appropriately represented so that the operator can exploit the information and abilities provided by the platform. Evaluating natural interaction from the perspective of the person interacting with the robot can be addressed afterwards.

Using an egocentric viewpoint (seeing the world from the robot's perspective) is usually better for navigating and avoiding obstacles, while an exocentric viewpoint (observing the world from an external perspective) gives the operator a better understanding of the environment's structure [Ricks *et al.*, 2004]. In previous work, we developed and evaluated interfaces with different viewpoints [Ferland *et al.*, 2009; Michaud *et al.*, 2010a]. We conducted a comparative study with 37 novice operators between 1) a video-centric display, 2) an augmented reality display (superimposing the video stream on a 3D virtual

model of the environment), and 3) a mixed-perspective display providing an exocentric viewpoint (the 3D model) in the center of the display and an egocentric view (the video feed), with a reference between both perspectives (the robot position) [Labonté *et al.*, 2010]. The second modality was revealed to be the most usable and most effective interface in terms of completion time and number of commands, especially for women, people older than 30 years old, or people working on computers less than 22 hr per week executing moving tasks. The third modality turned out to be preferable in situations requiring precise navigation but is sensitive to the robot's localization accuracy, which influences the alignment of the 3D model with object features in the video stream. In a follow-up pilot study, we experimented with an adjustable ego/exocentric 3D display, allowing operators to change viewpoints as desired [Ferland *et al.*, 2009]. From a test population of 13 participants, this functionality was found to be very useful, and experienced videogamers performed better, making the robot move at maximum speed. The growing popularity of videogames with the general population is certainly a factor to take into consideration in the design of teleoperation interfaces. In 1998, 3% of the Canadian population over 15 years of age spent an average of 1 hr 48 min per week playing video games. In 2010, these figures grew to 6% and 2 hr 20 min [Béchard, 2011].

Therefore, we began our exploration of IRL-1 as a telepresence platform by extending our work on the adjustable ego/exocentric 3D display [Ferland *et al.*, 2009] and investigating the added benefits and challenges of omnidirectionality and directed perception. This interface combines a 3D reconstruction of the environment, using either laser range finder readings, with a video projection method or a point cloud (produced by a stereoscopic camera or a Kinect sensor), and it allows the operator to switch seamlessly between egocentric and exocentric viewpoints. Viewpoints changes are rendered by smooth interpolation of both the position and orientation of the virtual camera. Exocentric viewpoints are necessarily limited by the camera's field of view (FOV). With the Kinect sensor mounted on IRL-1's orientable head, 3D video projection no longer had to stay aligned with the front of the robot : it could change according to the orientation (pan and tilt) of the head. Figure 3.4 illustrates the concept : (a) is the case when the virtual camera is aligned with the robot's orientation and head, located slightly behind the robot to visualize a representation of robot as in Nielsen et al. [2007] and Ferland et al. [2009] ; (b) represents the case of an exocentric viewpoint that is independent of the robot's head orientation, and (c) is when the virtual camera is aligned with the robot's head but not necessarily with the robot's orientation. This capability keeps the virtual camera aligned with the Kinect FOV, thus simplifying the virtual camera's pan and tilt control. The virtual target

is always at a 5 m distance from the robot, but the distance between the virtual camera's position and the robot can be adjusted, effectively acting as zoom control.

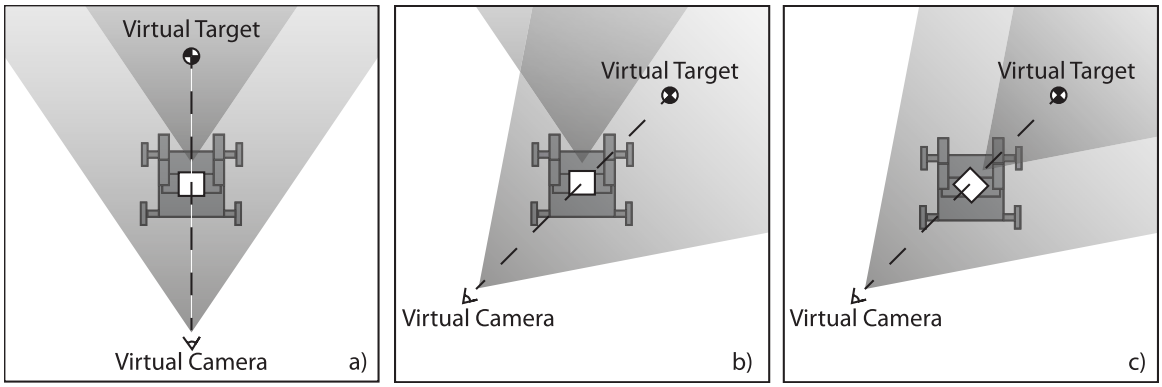


Figure 3.4 Top view of possible exocentric viewpoints : a) viewpoint aligned with the robot's orientation and head position ; b) viewpoint independent of the robot's orientation and head position ; and c) viewpoint aligned with the robot's head position but not with the robot's orientation.

To rapidly prototype and evaluate this concept, we implemented a simple experimental setup using rviz, which is the visualization tool provided with Willow Garage's Robot Operating System (ROS) [Quigley *et al.*, 2009], as shown in Figure 3.5. An articulated, semi-transparent model of IRL-1 is rendered at the bottom. Laser range readings are displayed as points with colors, from red to blue, based on the distance of the perceived objects along the robot's X axis. The point cloud generated by and visible from the Kinect camera is shown in the middle of the screen. Sounds located by ManyEars are displayed as red arrows pointing in the direction of the sound sources. The feasibility study involved teleoperating the robot in the cluttered environment, given no specific objectives, using one of two modes :

- **Mode 1** : IRL-1's head is always oriented forward, with no pan or tilt controls, but the virtual camera's position and target can be rotated around the robot's base, as in Figure 3.4(b). This mode corresponds to our previous work on adjustable ego/exocentric 3D displays [Ferland *et al.*, 2009].

- **Mode 2** : The virtual camera's position and target still rotate around the robot's base, and its head orientation follows, within its limits, the virtual camera's orientation, as in Figure 3.4(c). If the virtual target is beyond a soft angle limit in any direction ($\pm 30°$), the robot's base reorients itself toward the virtual target. This new display modality exploits the added

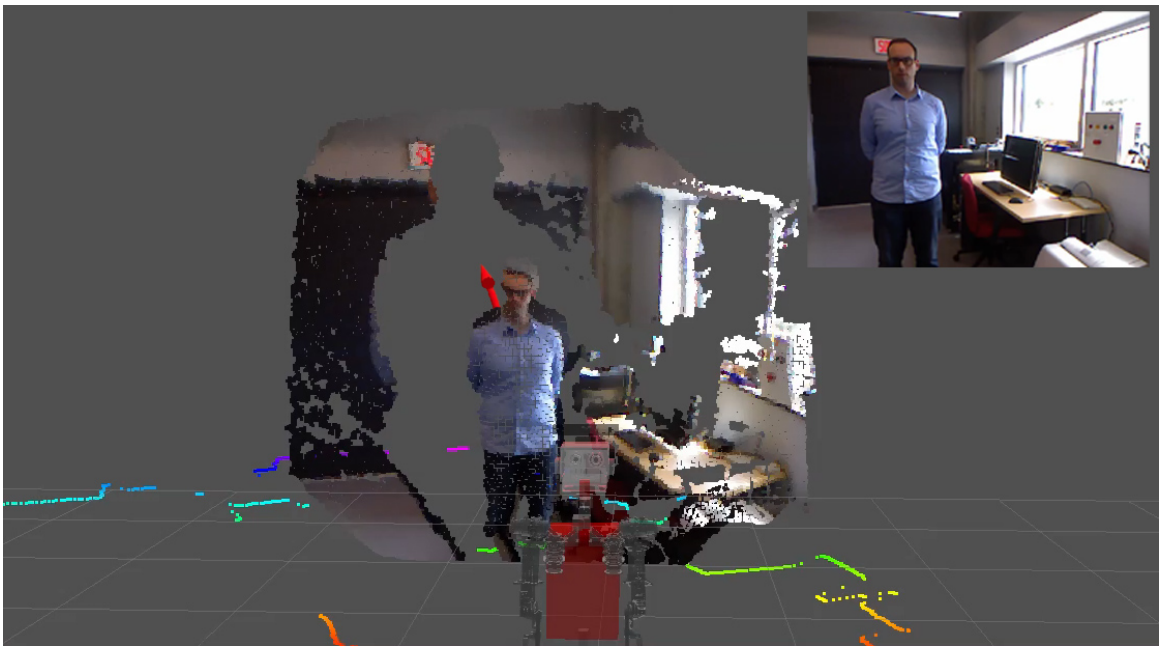


Figure 3.5 Telepresence prototype interface within rviz. The image captured by the Kinect camera is shown in the top right corner. The red arrow, partially masked by the Kinect point cloud, shows the direction of a detected sound source.

capabilities of IRL-1 by associating the user's virtual camera with the robot's motor control.

A convenience sample of seven male engineers, aged 22 to 35 years, teleoperated IRL-1 using each mode for up to 10 minutes.

From these experiments, we observed that Mode 2 has considerable potential. Orienting the Kinect while the base is moving helped locate nearby obstacles, especially those that could not be detected by the laser range finder because of its height. In Mode 1, objects on the ground at close range were not part of the Kinect's FOV, thus increasing the importance of the laser range readings for navigation. On the other hand, latency in the Kinect data feed became apparent when the pan-tilt unit state was updated on the display slightly earlier than the Kinect's point cloud. The Kinect data stream used considerable bandwidth and was subject to large variations in network latency for each point cloud. The result was that the Kinect point cloud was displayed temporarily out of alignment with the robot's head. In our implementation, no specific precautions were taken to minimize the effect of the network latency's variation when executing user commands and displaying perceived information. As mentioned in Kelly et al. [2011], teleoperation becomes even more difficult in environments surrounded by dense obstacles, with network latency and high speed

requirements. While IRL-1's maximum velocity is not significantly higher than in our work on the adjustable ego/exocentric 3D display (0.5 m/s vs. 0.3 m/s), IRL-1's head motion produces rapid changes to the content of the video feed and the virtual camera's position that need to be compensated for. Motion prediction with a velocity-driven dynamic model [Kelly *et al.*, 2011] and wave variables [Munir et Book, 2002] is currently being considered to compensate for latencies that can occur between graphical representation of the robot's state, and perception. This would enable rendering the environment perceived by the robot at a fixed rate, the model predicting parts of the state that have been delayed or lost.

We also learned that customization options should be available to the user. For instance, the threshold angle that triggers the robot's base movement when adjusting the pan angle of the virtual camera could be provided as a movement sensitivity setting. A participant complained that his commands, which were frequently very close to this movement threshold, resulted in jerkiness in both the robot's movement and the camera display. Allowing users to set this threshold would be preferable. Finally, an interesting idea for a test case would be to have participants engage in a teleoperated pursuit task and to measure the number of collisions, the distance between the robot and the target, and the time that the target remains visible on the display.

3.5 Computing Resource Management Feasibility Study

The design of a highly sophisticated mobile robot requires the integration of capabilities such as navigation, localization and mapping, vision and audio processing, tracking and recognition, graphical or natural interfaces, and planning and reasoning using different abstraction levels [Michaud *et al.*, 2007a]. Integrating many capabilities onto the same platform increases the level of complexity for controlling the robot and, incidentally, the load on the robot's computing resources. While technological progress in computer design expands the limits every year, we can also safely assume that software requirements will continue to increase in complexity for improved perception and reasoning capabilities. The use of offboard computing resources, such as web-enabled robots [Tenorth *et al.*, 2011] or cloud computing robotics, is an option. However, this approach is subject to latency and reliability issues, and its use is limited to locations with Internet connectivity.

Humans solve the problem of limited computing resources by selective attention. In psychology, selective attention has been studied according to two schools of thought [Bundesen et Habekost, 2004] :

- Early selection theory [Broadbent, 1958] proposes that human perceptual capacities are limited, and unimportant stimuli need to be filtered before they overload our analyzing capabilities.

- Late selection theory [Deutsch et Deutsch, 1963] proposes that our perceptual analysis capabilities are actually unlimited and that selection occurs later based on the perceived importance of the stimuli.

More recent work by Lavie 1995 attempts to unify both theories by showing that while our perceptual resources are limited, we try to exploit them to their maximum. Early selection would then only be necessary under a significant perceptual load.

We are currently investigating the implementation of selection attention mechanisms in our Hybrid Behavior-Based Architecture (HBBA) [Michaud *et al.*, 2010b], which is illustrated in Figure 3.6. Driven by data processed by Perception Modules, Behavior-Producing Modules (BPMs) are distributed processes used to generate commands sent to the robot's actuators using an action selection strategy (e.g., priority-based) [Pirjanian, 2000]. BPMs are activated and configured according to the Intentions of the system and evaluated in the Intention Workspace from concurrent Desires generated by the Motivation modules. Similar to BPMs, Motivations are distributed processes manifesting Desires for the accomplishment or the termination of Intentions, which are associated with the activation and configuration of BPMs and with the modulation of Perception Modules. Intentions are therefore used for early selective attention by activating or deactivating filters (represented by the $\bowtie$ symbol at the input of two Perception Modules) that modulate the flow of information from sensors to perceptual modules, or Late Selection by activating and configuring BPMs.

In a previous experimental setup to validate this framework, IRL-1 was programmed to track the 3D position of a blue ball with a Kinect camera and a tracking algorithm implemented with the Point Cloud Library (PCL) [Rusu et Cousins, 2011]. Meanwhile, a person was trying to distract IRL-1 by hand clapping or by speaking loudly. In this case, Study is the Motivation Module making the robot track the blue ball (using the Ball Localizer Perception Module and the Track Object BPM), while Interact makes the robot responsive to sounds by orienting its head toward the sound sources (using ManyEars for Sound Localizer and the Track Sound BPM). To test our early selective attention mechanism, we constrained the available CPU resources so that these BPMs could not be executed concurrently in real time : either sound sources were detected after a delay of multiple seconds, or the ball was tracked at rates of less than 1 Hz. A less computer-

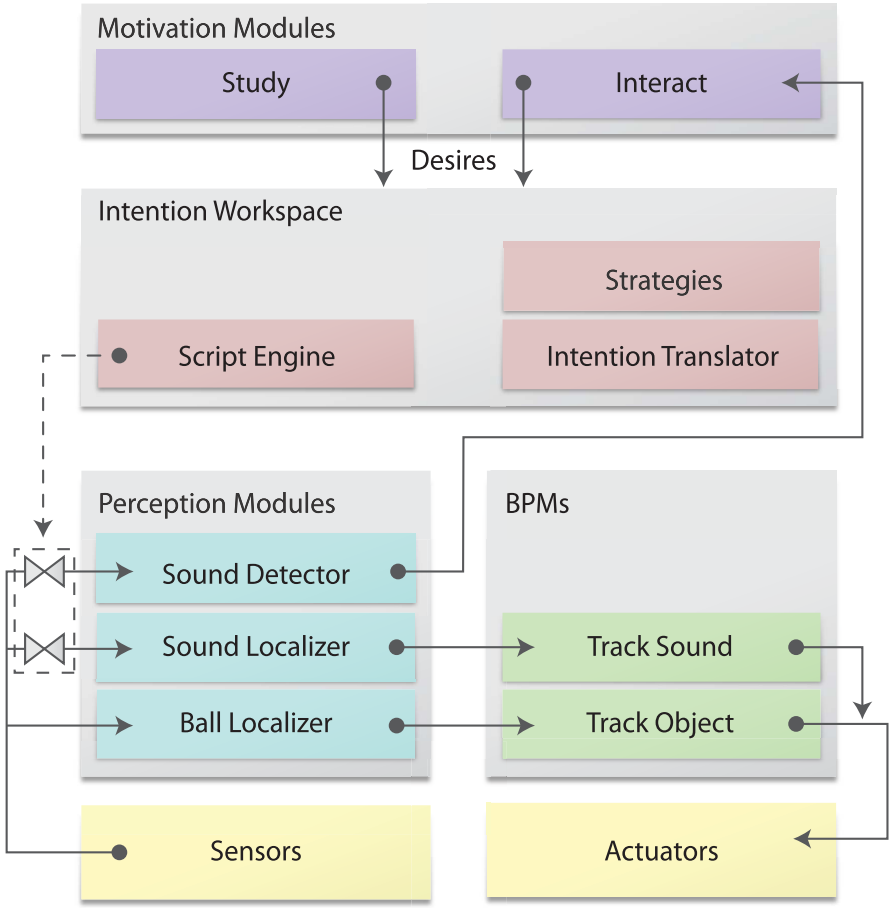


Figure 3.6 HBBA conceptual framework with experiment-specific modules.

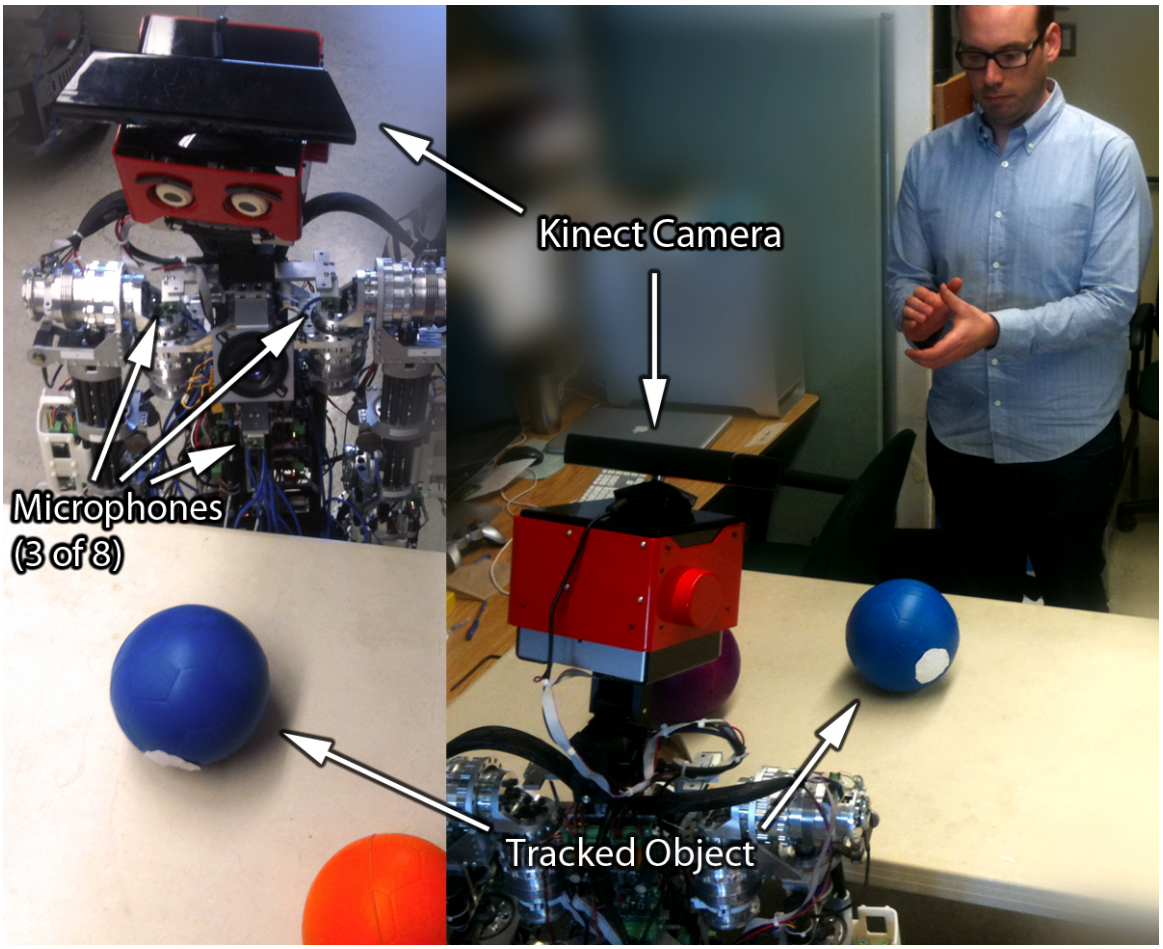


Figure 3.7 Experimental setup for the feasibility study on resource management.

intensive Perception Module called Sound Detector was therefore added to monitor loud noises using a single microphone located on the front of the torso. Unlike Sound Localizer, Sound Detector can stay activated when no loud noise is perceived without consuming significant resources, allowing IRL-1 to be more responsive while tracking the blue ball. When a loud noise was detected, the average delay between the beginning of the hand clap sound and the movement of the head pan-tilt unit was 1015 ms ($\sigma = 294$ ms), compared to 692 ms ($\sigma = 122$ ms) when Sound Localizer was the only running process. This additional 323 ms delay was mostly caused by the time required by ManyEars to process the audio stream buffer of the last recorded second to locate the sound source.

To determine if people interacting directly with the robot would consider the additional delay acceptable, we asked a convenience sample of eight male engineers (aged 22 to 35 years) to distract the robot and to indicate when a) a delay is noticeable, b) the delay is considered uncomfortable, and c) the delay is considered unacceptable. Artificial

delays from 0 to 3000 ms with 100 ms increments were generated and added to the base response time. Participants were asked to base their evaluation on their own expectations of the robot and of the task, and not to compare it to a human. Before conducting the experiments, participants were allowed to interact with IRL-1 to find the required sound level for reliably distracting its attention from the blue ball.

Figure 3.8 illustrates the results. Noticeable additional delays are mainly identified between 400 and 600 ms (for 7 out of 8 participants), while for delays considered uncomfortable and unacceptable the ranges are much larger. This suggests that the overhead delay of 323 ms is not a variable that should influence natural HRI experiments.

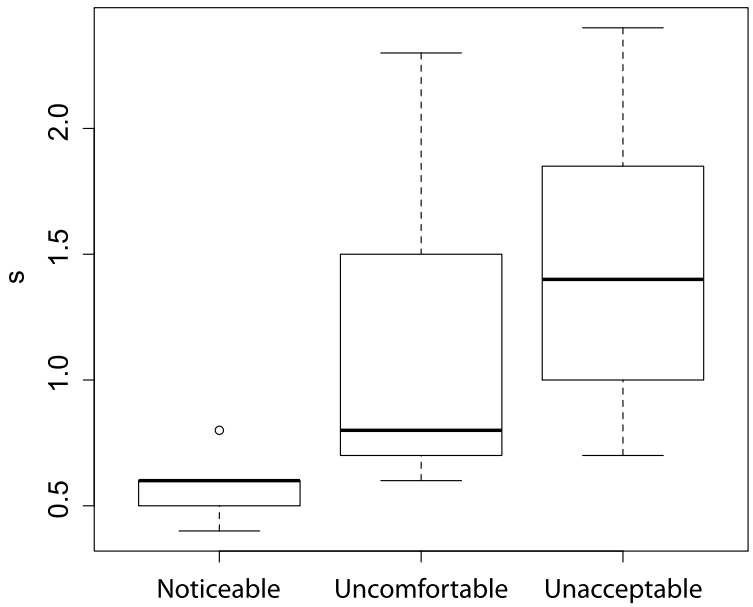


Figure 3.8 Noticeable, uncomfortable and unacceptable delays for IRL-1's responsiveness to sound distraction.

Comments from participants provided interesting insights for the preparation of future experiments. For instance, when hand clapping was not loud enough to be detected, a participant took this as a sign that the robot was busy and that the ignored interruption was not simply caused by a malfunction. In that regard, IRL-1's facial expressions revealed an interesting side effect. Facial expressions on IRL-1 were programmed to switch between variations of a single emotion every five seconds. This results in IRL-1's eyes moving slightly at regular intervals, which caused some participants to ask if this should be interpreted as a sign of IRL-1 being successfully distracted by the sound. They were told

that it was not the case because IRL-1's head had not moved, but some participants indicated that it made the robot appear more responsive. This may be a simple and natural gesture for making the robot respond rapidly to the detection of a sound before ManyEars can provide a location for the detected sound. Such an addition would be in line with the observation that external observers feel that an *attentive* robot, which moves its head toward the interlocutor in a two-person, turn-taking conversation, is considered to be more natural than a *distracted* robot (moving its head with a 500 ms delay) [Trafton *et al.*, 2008]. Finally, a participant commented that he would have said a much lower delay was unacceptable if he had been trying to disturb a person instead of a robot ; he explained that he did not really expect contemporary robots to react to his interruptions. It may therefore be interesting to investigate people's expectation for interaction responsiveness in relation to a robot's appearance, capabilities, and role.

3.6 Discussion and Future Work

The HRI perspective addressed in this paper relates to the concept of "natural interaction design", which we could define as the practice of designing technologies and capabilities for human-robot interaction. Pursuing the long-term objective of developing robots, algorithms, and technologies for natural HRI, experiments conducted over the years in different HRI contexts (e.g., children, home care assistance, interactions in public spaces) have led us to design new technologies, such as DEAs and ManyEars, that have been integrated into IRL-1, an omnidirectional, force-sensing, compliant humanoid robot. The feasibility studies presented in this paper are part of an iterative and incremental process aimed to improve understanding of how the integration of the advanced motion, perception, and processing capabilities influence each other in natural interaction. Experimenting with new technologies brings new challenges, and we believe that the feasibility studies conducted with IRL-1 are an important stepping stone to evaluating design and evaluation considerations for safe and efficient progress toward more advanced integration and HRI studies.

The feasibility studies explored distinctive capabilities of IRL-1 : the ability to be physically guided by perceiving forces through DEAs used for active steering of the omnidirectional platform ; the integration of vision, motion and audition for an augmented telepresence interface ; and the influence of delays as computationally expensive processes (such as ManyEars) are integrated in responding to user requests. In addition to demonstrating how these capabilities can be used in HRI, this paper illustrates how their design and evaluation are intrinsically interrelated. For instance, the contact point at which IRL-1 is physically guided influences interaction, and this should be taken into consideration

both for the design (by also using the arms' DEAs to measure applied forces) and the evaluation (by observing where users guide the platform). The orientation of IRL-1's head (and therefore the Kinect), independent of the platform motion, can make network delays visible in the graphical display, making this an important consideration in the design and evaluation of telepresence interfaces for humanoid robots. And as reasonable delays can be reached using a selective attention mechanism as computationally expensive processes are integrated on IRL-1, simple but responsive facial gestures may be beneficial in natural interaction, and should be investigated further.

As the design of IRL-1 progresses, many additional experiments can be imagined, such as the use of facial expressions to communicate information about motion constraints in physically guiding the platform ; the study of user preferences in controlling mobility and viewpoints and of representing forces and sound classification on the graphical display ; and vocal interaction with IRL-1 as it navigates in a cluttered environment (which requires the addition of SLAM capabilities, which are computationally expensive), while responding in a timely fashion to environmental conditions and user requests. In developing new technologies and conducting trials in such a wide set of conditions, we believe that advancing the state of the art in natural HRI is an interesting avenue to address the fundamental integration challenge in the field of artificial intelligence [Brachman, 2006], making robots more versatile, meaningful, and rational machines.

Acknowledgments

This work is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canada Research Chair (CRC) in Mobile Robotics and Autonomous Intelligent Systems, and the Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT).

CHAPITRE 4

Perceptual Filtering for Selective Attention in a Behavior-Based Robot Control Architecture

Avant-propos

Auteurs et affiliation :

F. Ferland : étudiant au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a développé la théorie et l'implémentation complète de HBBA et de l'attention sélective présentée dans cet article.

François Michaud : professeur, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique. Il a contribué à la conception de l'architecture HBBA et à la révision de cet article.

Date de soumission : 2 octobre 2014

État : En cours de révision

Revue : Autonomous Robots

Titre français : Filtrage perceptif pour l'attention sélective dans une architecture comportementale pour robots

Contribution au document : Cet article contribue à la thèse en venant valider l'implémentation du filtrage perceptif en tant que mécanisme d'attention sélective. Le mécanisme y est décrit en détails, ainsi que son impact positif sur la charge de calcul dans une tâche d'interaction avec le robot IRL-1.

Résumé en français : Les robots autonomes modernes doivent intégrer multiples modalités perceptives et comportementales pour être utiles dans la vie de tous les jours. Une telle intégration est contrainte par la capacité limitée des ordinateurs embarqués des plateformes domestiques en robotique. Pour répondre à ce problème, le filtrage perceptif, un mécanisme d'attention sélective, peut être utilisé pour gérer efficacement les ressources informatiques du robot en fonction de ce qu'il a à accomplir. Cet article décrit notre

implémentation du filtrage perceptif dans une architecture de contrôle comportementale pour robot utilisant ROS et comment il peut être utilisé pour optimiser dynamiquement l'utilisation des ressources en calcul disponible sur le robot. Notre mécanisme de filtrage perceptif est démontré et validé avec une plateforme humanoïde mobile qui intègre une navigation autonome et téléopérée, de la reconnaissance de codes QR et de visages, et la localisation de sources sonores.

Abstract

Modern autonomous robots must integrate multiple perceptual and behavioral modalities to be useful in our daily lives. Such integration is constrained by the limited computing capacity of domestic-scale robotic platforms. To alleviate this issue, perceptual filtering, a selective attention mechanism, can be used to efficiently manage computing resources based on what the robot has to accomplish. This paper describes our implementation of perceptual filtering in a behavior-based robot control architecture, implemented using ROS, and how it can dynamically optimize the use of the computing resources available on the robot. Our perceptual filtering mechanism is demonstrated and validated using a mobile humanoid platform integrating autonomous and teleoperated navigation, QR-code recognition, face recognition and sound localization capabilities.

Keywords : Selective attention, Robot control architecture, Resource management

4.1 Introduction

Over the past few years, complex autonomous robotic systems, such as the entries of the 2007 DARPA Urban Challenge [Montemerlo *et al.*, 2008; Urmson *et al.*, 2008] or more recent efforts by Mercedes-Benz [Ziegler *et al.*, 2014] and Google [Guizzo, 2011], have shown great advances in terms of hardware and software integration. However, domestic-scale mobile robots have much more limited capabilities and more stringent resource allocation requirements [Hawes *et al.*, 2007; Michaud *et al.*, 2007a]. And while technological progress in computer design pushes away these limits every year, it can be safely assumed that software requirements will continue to grow to provide robots with more and more perception and reasoning capabilities. Integration of Simultaneous Localization and Mapping (SLAM), object manipulation and people recognition is to be expected on modern robots. Simultaneously processing all these capabilities requires computing resources that can exceed what is available on service robots. As an alternative, providing access to offboard computing resources (e.g., web-enabled robots [Tenorth *et al.*, 2011], ubiquitous, cloud-based robotics [Chibani *et al.*, 2013]) is possible, but can be subject to latency and reliability issues, and limits its use to locations with Internet connectivity.

Humans solve the problem of limited computing resources through selective attention. In psychology, selective attention has been studied according to two schools of thought [Bundesen et Habekost, 2004] :

- The Late Selection (LS) theory [Deutsch et Deutsch, 1963] suggests that our perceptual analysis capabilities are actually unlimited and that selection occurs later based on the perceived importance of the stimuli.

- The Early Selection (ES) theory [Broadbent, 1958] states that human perceptual capabilities are limited, and unimportant stimuli need to be filtered before they overload our processing capabilities.

Recent work [Lavie, 1995, 2005] attempt to unify both theories by showing that while our perceptual capabilities are limited, we try to exploit them to their maximum : ES would then only be necessary under a significant perceptual load.

Attention mechanisms have already been implemented on robots, and more specifically for visual processes [Breazeal et Scassellati, 1999; Demiris et Khadhouri, 2006; Ude *et al.*, 2007; Vijayakumar *et al.*, 2001]. Additionally, robot control architectures have mechanisms that can be associated to selective attention. For instance, behavior-based systems [Mataric et Michaud, 2008], with the activation of behavior modules, is one approach that can be associated with LS. Scheutz and Andronache [Scheutz et Andronache, 2004] present a generic architectural mechanism for dynamic behavior selection, allowing the robot to direct its attention to perception associated with the activated behaviors. An example of ES is the CoSy Architecture Schema (CAS) and the CAS Toolkit (CAST) [Hawes *et al.*, 2007; Hawes et Wyatt, 2010]. CAS consists of a collection of processing components that can exchange information via a shared working memory. Instead of broadcasting information to all components, CAST has a mechanism to select the components that require the information, limiting processing of irrelevant information by the processing components of the architecture.

Such selective attention mechanisms are based on all-or-nothing policies : the processing modules are either activated or deactivated, or the information is either available or not. A novel approach to manage computing resources of robot control architectures would be to adapt the rate at which the stimuli reach the processing modules, which we refer to as perceptual filtering. For distributed software frameworks employing message passing as their communication mechanism (such as ROS [Quigley *et al.*, 2009]), perceptual filtering can be achieved by altering the message passing rate using time-decimating frequency filters on the input channels of processing modules. This paper presents how perceptual filtering can be used in a behavior-based robot control architecture to avoid overloading the computing resources of a mobile robot that integrates SLAM, QR-code recognition, face recognition and sound localization capabilities.

The paper is organized as follows. Section 4.2 presents the robot platform and control architecture used to validate our perceptual filtering mechanism. Section 4.3 describes how modules in the architecture can be characterized from a computing resources consumption standpoint, and what extensions are added to the behavior-based architecture to implement perceptual filtering. Section 4.4 then presents experimental results to illustrate how the mechanism works, and comparing computing resources consumption with and without the use of perceptive filtering on a real robot. Finally, Section 4.5 provides observations about the results and design choices made.

4.2 Robot Platform and Control Architecture

The platform used for this work is IRL-1/TR, shown by Fig. 4.1. IRL-1 is a humanoid torso equipped with an expressive face, an orientable head, two compliant arms with four degrees of freedom (DOF) and grippers [Ferland *et al.*, 2012]. IRL-1/TR has the torso installed on a differential-drive mobile base [Michaud *et al.*, 2010a]. IRL-1/TR provides a large range of sensors, such as :

- A wireless gamepad from Logitech, featuring ten individual buttons and two analog joysticks sampled at a rate of 10 Hz.
- A color USB camera with a panoramic lens from Immervision (IMV). It provides 1024×768 RGB images at 10 Hz of the full hemisphere in front of the robot.
- A Microsoft Kinect sensor mounted on the head of the robot. It provides 640×480 RGB-D images at 15 Hz with a $57° \times 43°$ field of view and depth measurements of up to 5 m.
- An array of eight omnidirectional microphones, located around the torso of the robot. Audio from these microphones is captured at 16-bit and 48 kHz.
- A Hokuyo UTM-30LX laser range finder situated on top of the mobile base, 40 cm from the ground. A scan consists of 720 equally spaced range and reflectance readings over 30 m on a 180° arc centred in front of the robot. Scans are generated at a rate of 40 Hz.

IRL-1/TR has two on-board computers : a 2.67 GHz second generation Intel Core i7 dual core processor with 4 GB of RAM located in the mobile base ; and a 2.10 GHz third generation Intel Core i7 quad core processor, featuring Hyper-Treading with 4 GB of RAM, located in front of the torso. The 2.67 GHz system hosts processes related to the control

of the mobile base, while the 2.10 GHz system is used for the torso controllers (arms, head expression and orientation), sensor processing, and the coordination of the robot control architecture.

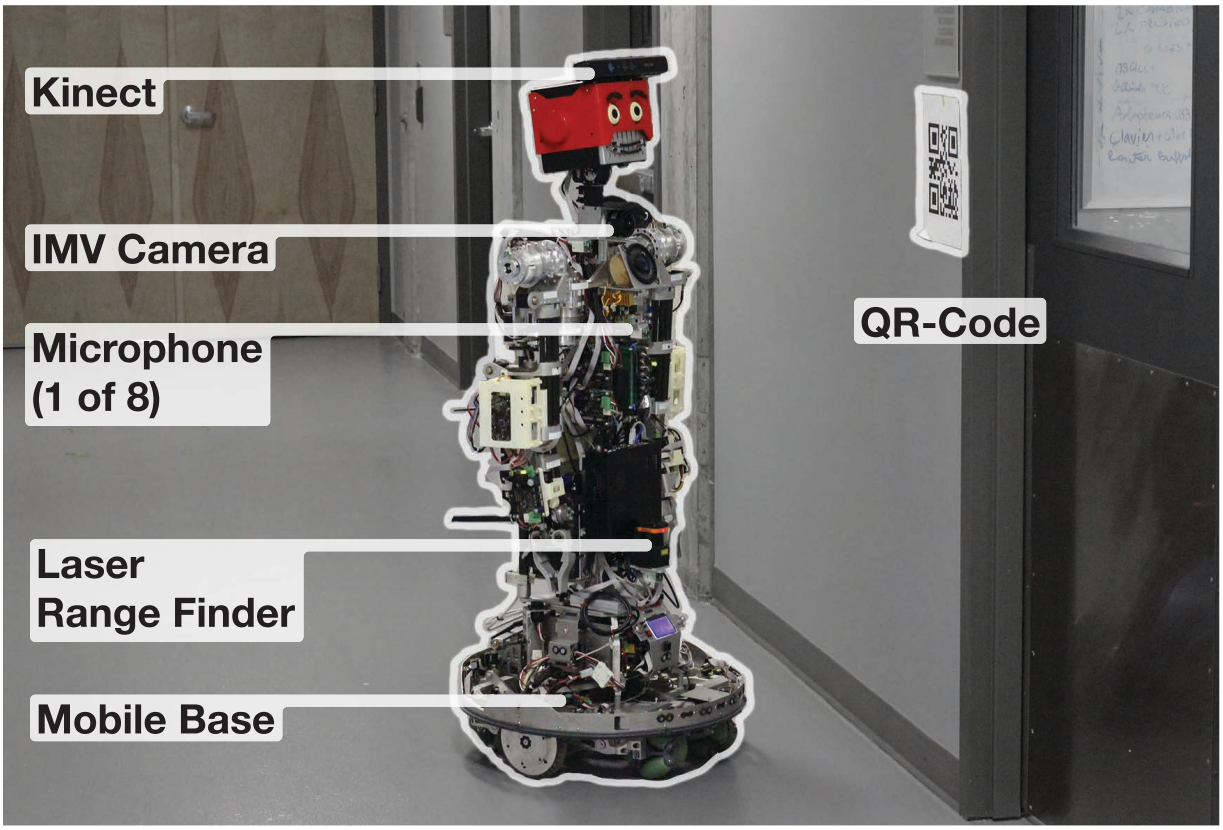


Figure 4.1 The IRL-1/TR robot, approaching a QR-code affixed next to a door.

One of the targeted applications for IRL-1/TR is to be used as a tour-guide of our facility. This requires IRL-1/TR to navigate in the environment, engage people, provide verbal descriptions at specific locations, and point to lab equipment, all in a robust and natural manner. To associate tour interactions with specific locations, IRL-1/TR locates pre-affixed QR-codes. Each QR-Code describes which tour interaction to perform at the location of the QR-code. To configure the tour, IRL-1/TR is teleoperated by an operator to map the environment and locate QR-codes, along with detecting other external events (loud noises and people faces) to allow interaction with people. This configuration task constitutes the application scenario in which our perceptual filtering mechanism is validated in this paper.

To implement the tour-guide configuration task, the Hybrid Behavior-Based Architecture (HBBA) [Ferland *et al.*, 2012; Michaud *et al.*, 2010b] is used to integrate a set of Perception modules using multiple sensing modalities (visual, audio and range finder-based), with two

Behavior modules controlling velocity of the mobile base. Figure 4.2 illustrates the modules implemented in HBBA for this application, as described in the following subsections.

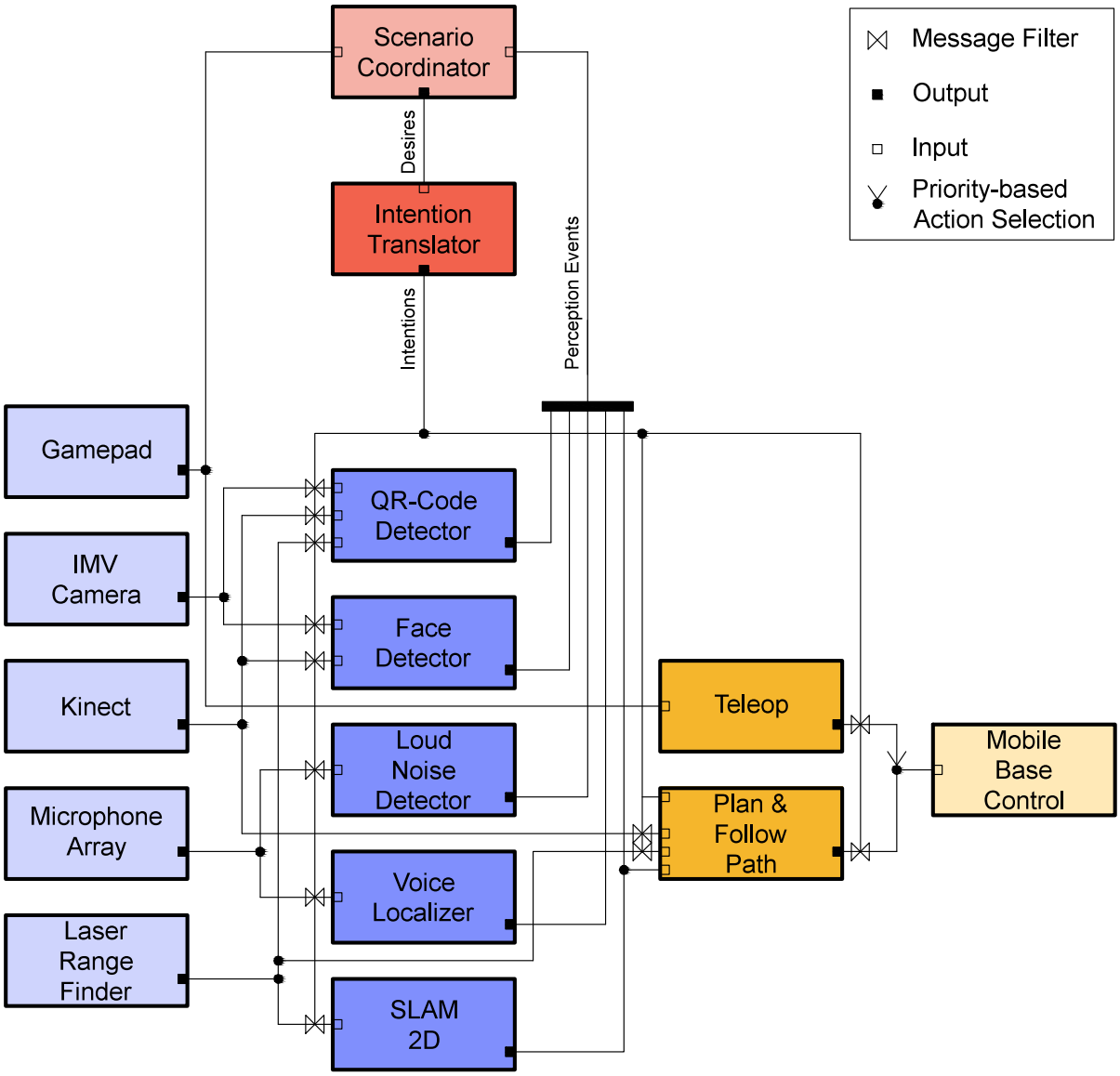


Figure 4.2 HBBA used with IRL-1/TR to implement the tour-guide configuration task. Sensor modules are in light blue, Perceptual modules in dark blue, and Behavior modules in dark yellow.

4.2.1 Perception Modules

QR-Code Detector is based on the open source ZBar library[1] and is configured to detect and decode QR-codes the size of a standard US Letter paper. It takes data from the IMV

[1]http ://zbar.sourceforge.net

camera for QR-codes located in its peripheral vision, and from the Kinect camera for QR-codes located in front of the robot. Proper localization of a QR-code can be approximated from the average distance measured by the intersection of the laser range finder data and the field of view of the IMV camera, or obtained by computing the average depth of the pixels in the detected region of the depth image provided by the Kinect. Decoding occurs reliably using the IMV camera when codes are approximately 1 m from the robot, and less than 2 m using the Kinect camera.

Face Detector takes data from the IMV camera and the Kinect to localize potential faces. Images from both cameras go through the same detection process using the OpenCV 2.4.5 implementation and the database of the Local Binary Pattern approach [Ahonen *et al.*, 2004]. When a face is detected using the Kinect, the average depth in the detection region is used to localize the person's face. However, because of the height and the limited field of view of the Kinect, a person's face may be outside of this range depending on its height and distance. This may cause frustration as people expect to be seen by IRL-1/TR when standing right in front of it, especially since its eyes are clearly visible. As a solution, the Face Detector module extends its field of view by also looking for faces in regions of the IMV image that potentially feature standing humans. The module uses a gaussian distribution of the targeted people's average height and the location of the IMV camera relative to the ground to estimate the distance (with standard deviation) of people's faces.

Loud Noise Detector monitors a single microphone (located in front of IRL-1/TR, right in the middle of the torso) to quickly detect loud noise events by processing single frames of audio data (512 samples at 48 kHz, or approximately 10.67 ms). It computes a moving average of the energy level over the last 200 frames of the sound signal. When the energy difference from the moving average is more than 20 times higher, this module produces a loud noise event, represented by a time stamp. This threshold was determined empirically to produce an event when a person calls IRL-1/TR by its name (using a normal conversation level) or claps its hands, both from a distance of 3 m or less. Furthermore, the module always perceives one or more invalid noise events at initialization, as the moving average initialize itself for the first 200 frames of signal (2.134 s).

Voice Localizer monitors separated sound sources provided by ManyEars [Grondin *et al.*, 2013], a sound source localization, tracking and separation algorithm using the 8-microphone array mounted on the torso of IRL-1. ManyEars performs real-time beamforming for localization, particle filtering for tracking, and Geometric Source Separation (GSS) to provide distinct audio streams of each sound source. It can localize up to four distinct sources, and separate the content of up to two of those. Voice Localizer analyzes

these sources to find the presence of sounds typically associated with human voices based on instant pitch extraction and classification [Sasaki *et al.*, 2009]. A database of known speech and non-speech audio frames is built using vector quantization. Then, new audio frames are classified according to their probability of belonging to one or both of these specific classes. To estimate the distance of sound sources, it is assumed that voices are always coming from a human being in a standing position. Like Face Detector, the module uses a gaussian distribution of the targeted people's average height and the location of the microphone array relative to the ground to estimate the distance (with standard deviation) of the mouth of the person talking.

SLAM 2D relies on the ROS implementation of OpenSlam's Gmapping SLAM algorithm [Grisetti *et al.*, 2007]. It relies on laser range readings to create a 2D occupancy grid and an estimation of the current pose of IRL-1/TR. The module is configured to maintain a 50 m $\times$ 50 m map with a cell resolution of 0.05 m.

4.2.2 Behavior Modules

Teleop is a simple teleoperation module that linearly transforms inputs from the wireless gamepad into velocity commands for the mobile base. Two analog joysticks are used : one for the forward/backward linear velocity, and the other for the angular velocity to make the robot rotate around its center. Commands can be generated only when the live-man button is pressed.

Plan & Follow Path is based on the ROS navigation stack[2] and takes navigation goals to generate velocity commands. The map obtained from the SLAM 2D module is used as a global occupancy map in its original size and resolution. The module monitors both the depth image of the Kinect and the data from the laser range finder to build a 4 m $\times$ 4 m local occupancy map with 2.5 cm resolution for static and dynamic obstacles. Planning is done in two phases : a path is determined with the A* algorithm on the global occupancy map, and velocity control is done using the Dynamic Window Approach (DWA) [Fox *et al.*, 1997] at a 10 Hz rate and exclusively using the local occupancy map.

4.2.3 Action Selection and Control

Mobile Base Control takes linear and angular velocity commands and transform them into rotational velocities for both wheels of the differential-drive mobile base. The velocity is limited to 0.40 m/s and 1.00 rad/s, and acceleration/deceleration to 0.40 m/s^2 and

[2]http ://wiki.ros.org/navigation/

4.00 rad/s^2. This module can receive commands from multiple Behavior modules, and performs priority-based action selection at a rate of 10 Hz. As shown in Fig. 4.2, in our implementation, Teleop always has priority over Plan & Follow Path.

4.2.4 Scenario Coordinator

To have the robot adapt its goals based on tasks to accomplish and situations experienced in its environment, HBBA [Ferland *et al.*, 2012; Michaud *et al.*, 2010b] adopts the following working principles :

- Behaviors can be activated and configured according to what are referred to as the Intentions of the robot.
- An activated behavior can provide commands to be executed by the Mobile Base Control module.
- High-level goals for the robot are generated and monitored by Motivation modules, which generate Desires for the satisfaction or inhibition of Intentions. For the tour-guide configuration task, *Scenario Coordinator* (SC) is the only Motivation module.
- Desires d generated by Motivation modules are instances of Desire classes k_d. A Desire instance d can be associated with only one Desire class k_d. A Desire class defines what the robot has to accomplish, and it can be configured by defining parameters, named p_d. For the tour-guide configuration task, the Desire classes are :
    - **FindQRCodes** indicates that QR-codes should be detected and decoded.
    - **LocateFaces** indicates that faces should be detected and localized.
    - **DetectNoises** indicates that loud noise events should be detected.
    - **LocateVoices** indicates that people voices should be detected and localized.
    - **SLAM** indicates that simultaneous localization and mapping should be performed.
    - **Teleoperation** indicates that the robot should be controllable by an operator.
    - **GoTo** indicates that the robot should go to location (x, y, θ), where θ is the desired orientation angle.

- Desires have an Intensity integer value $i_d \in [0, t_i]$ that represents the importance of fulfilling the desire, t_i being the maximum value.

- Desires have Desired Utility values $u_d \in [0, t_u]$. Desired Utility is a parameter that quantifies the minimum degree of fulfillment expected by a Desire. It is an non-negative integer, with 0 meaning that a Desire must inhibit a Desire class, and t_u representing the maximum value.

- Desires also have Security parameters $z_d \in \{0, 1\}$ to indicate whether ($z_d = 1$) or not ($z_d = 0$) it is critical that the desire must be fulfilled in priority (e.g., for security reasons or to fulfill essential goals).

When IRL-1/TR is initialized, *Scenario Coordinator* (SC) produces five Desires instances :

1. *SC_FindQRCodes* to record QR-codes encountered ($p_1 = \{\}$, $i_1 = 8$, $u_1 = 1$, $z_1 = 0$).

2. *SC_LocateFaces* to identify people that are passing by ($p_2 = \{\}$, $i_2 = 8$, $u_2 = 1$, $z_2 = 0$).

3. *SC_DetectNoises* to detect if someone is trying to get the robot's attention, for instance by talking or by clapping its hands ($p_3 = \{\}$, $i_3 = 8$, $u_3 = 1$, $z_3 = 0$).

4. *SC_SLAM* to map the environment and eventually be able to plan a path in it ($p_4 = \{\}$, $i_4 = 8$, $u_4 = 1$, $z_4 = 0$).

5. *SC_Teleoperation* to allow an operator to guide IRL-1/TR ($p_5 = \{\}$, $i_5 = 8$, $u_5 = 1$, $z_5 = 1$).

Scenario Coordinator also produces two additional Desires in specific conditions :

6. *SC_LocateVoices* when a loud noise event is detected ($p_6 = \{\}$, $i_6 = 8$, $u_6 = 1$, $z_6 = 0$). This Desire is regenerated every 5 sec as long as an audio frame is properly localized, to try to capture in one continuous audio stream what someone is saying.

7. *SC_GoTo* to make IRL-1/TR go back to its initial location when Gamepad button 3, situated at the top right, is pressed ($p_7 = \{0, 0, 0\}$, $i_7 = 8$, $u_7 = 1$, $z_7 = 0$).

Intensity values are all equal to t_i, which is set to 8 to match the range of Utility values.

4.2.5 Intention Translator

The Intention Translator module monitors and arbitrates the Desires generated by Motivation modules. In HBBA, the Intention Translator module is designed to handle a large set of Desires coming from multiple Motivation modules. In the following, we illustrate the mechanism using the tour-guide configuration task, which only has one Motivation module. This constitutes a simple but sufficient case scenario compared to what the Intention Translator module can handle.

The Intention Translator takes Desires and combine them in the following matrices, with D rows coresponding to the number of Desires, and K columns for the number of available classes :

- $P_{D\times K}$, the Parameter matrix.
- $I_{D\times K}$, the Intensity matrix. Undefined values are identified by zero.
- $U_{D\times K}$, the Desired Utility matrix. Undefined values are identified by $\emptyset$.
- $Z_{D\times K}$, the Security matrix. Undefined values are identified by empty sets.

Since each Desire can only be instantiated from a single Desire class, each row of matrices P, I, U, and Z contains a single non-zero item.

Desires describe what to achieve, but not how to achieve them. The role of the Intention Translator module is to activate and configure the Perception, the Behavior and the Action Selection and Control modules according to what to achieve. This is referred to as the Intentions of the robot, and are derived from Strategies. A Strategy s is defined in relation to a Desire class k_s by the following :

- $v_s \in]0, t_v]$, its Produced Utility, i.e., the degree of usefulness a Strategy has when fulfilling a Desire, with t_v setting the maximum level.
- l_s, its dependencies (or links) to other Desire classes. For instance, a Strategy for **GoTo** can require that a **SLAM** desire be also fulfilled.
- $\phi_s = \{\phi_a, \phi_b, \phi_c, ...\}, \phi \in \{1, \infty\}$, configuring a subset of the message filters of the robot control architecture. In HBBA, message filters are employed to control the flow of messages between modules. These filters are shown by $\bowtie$ symbols in Fig. 4.2. A message filter performs time-based decimation of the signal by letting one message pass through for every ϕ messages received : $\phi = 1$ lets all messages go through, while a value of $\phi = \infty$ blocks all of them, implementing an all-or-nothing policy.

By default, $\phi = \infty$, and each Strategy describes the message filters with $\phi = 1$, thus allowing messages to reach the modules to activate to fulfill the associated Desire class. For instance, behavior activation is implemented by using message filters between the command output of a Behavior module and the input of an Action Selection and Control module, setting $\phi = 1$ when the behavior is activated, and $\phi = \infty$ when deactivated.

For the tour-guide configuration task, t_u and t_v are also arbitrarily set to 8. Inputs and outputs filters of a module affected by Strategies are referred to following this convention : "$\phi_{[x-]ModuleName[-y]}$", where x refer to all inputs of the module, and y to all of its outputs. In this paper, Strategies affect the same ϕ for all the inputs and/or outputs of a single module. There are Strategies for each Desire class k :

1. **FindQRCodes** : $v_s = 8$; $l_s = \{\}$, $\phi_s = (\phi_{x-QR-CodeDetector} = 1)$, with x referring to all stimuli inputs of the QR-Code Detector module.
2. **LocateFaces** : $v_s = 8$; $l_s = \{\}$, $\phi_s = (\phi_{x-FaceDetector} = 1)$.
3. **DetectNoises** : $v_s = 8$; $l_s = \{\}$, $\phi_s = (\phi_{x-LoudNoiseDetector} = 1)$.
4. **LocateVoices** : $v_s = 8$; $l_s = \{\}$, $\phi_s = (\phi_{x-VoiceLocalizer} = 1)$.
5. **SLAM** : $v_s = 8$; $l_s = \{\}$, $\phi_s = (\phi_{x-SLAM2D} = 1)$.
6. **Teleoperation** : $v_s = 8$; $l_s = \{\}$, $\phi_s = (\phi_{Teleop-y} = 1)$, with y referring to the output of the Teleop module.
7. **GoTo** : $v_s = 8$; $l_s = \{$**SLAM** : 1$\}$, adding a implicit **SLAM** Desire, $\phi_s = (\phi_{x-Plan\&FollowPath-y} = 1)$, referring to both stimuli and output of the Plan & Follow Path module.

Intention Translator takes Desire instances as inputs to select Strategies and generate Intentions expressed in terms of Φ', the set of all message filters, with the set of configuration parameters P' of the processing modules. To determine which Strategies to select, Intention Translator solves a linear Constraint Satisfaction Programming (CSP) [Russel et Norvig, 2003] problem. Specifically, the goal is to determine the Strategy Selection vector $A_{S\times 1}$, where $a_s \in \{0, 1\}$ indicates whether Strategy s is selected (1) or not (0). This selection process occurs each time the Desire set changes, which leads to changes of Φ' and P'. Figure 4.3 represents the Strategy selection process, and each step is described in the following subsections. To illustrate this process, a simple example is used with only two Desire classes : **SLAM** ($k = 1$) and **GoTo** ($k = 2$) ; and two conflicting instances of **GoTo**

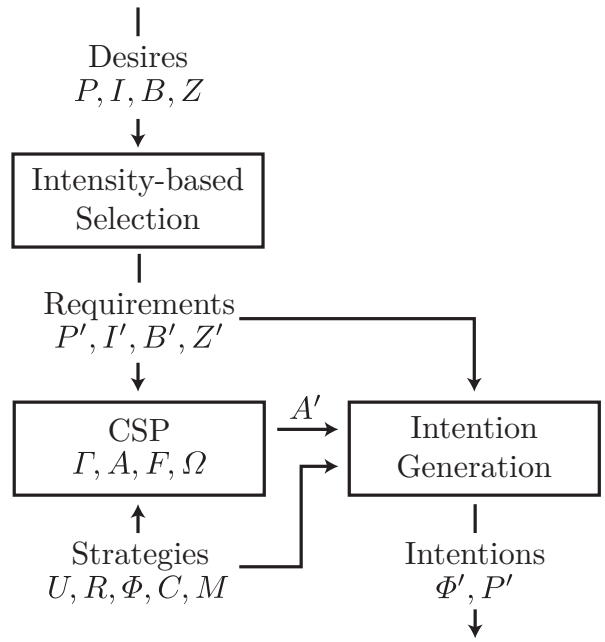


Figure 4.3 Strategy selection process, starting from instances of Desires (P, I, B, Z) to the generation of Intentions (Φ', P'), with extensions added for perceptual filtering (C and M).

Desires : 1) to go 10 m forward with $i_1 = 4$, and 2) to go 10 m to the left with $i_2 = 8$. Setting Desired Utilities to 1 signifies that the Desire can be fulfilled using a Strategy with the lowest v. Values for t_i, t_u and t_v are the same as the tour-guide configuration task. The following matrices define the illustrative example :

$$P = \begin{bmatrix} \{\} & \{10, 0, 0\} \\ \{\} & \{0, 10, 0\} \end{bmatrix}, I = \begin{bmatrix} 0 & 4 \\ 0 & 8 \end{bmatrix}, U = \begin{bmatrix} \emptyset & 1 \\ \emptyset & 1 \end{bmatrix}, Z = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{4.1}$$

Intensity-based Selection

Since multiple Desires can be defined simultaneously for a single Desire class, Intensity-based Selection consists of choosing the instance d of Desire class k_d with the highest Intensity value. This builds the requirements to be satisfied by the CSP algorithm, and defined four vectors : the desired parameters $P'_{K\times 1}$ of the Desires ; their desired Intensity $I'_{K\times 1}$; their desired utilities $U'_{K\times 1}$; and the desired security status $Z'_{K\times 1}$. For the illustrative example, Desire 2 has a higher Intensity value than Desire 1, which results in :

$$P' = \begin{bmatrix} \{\} \\ \{0, 10, 0\} \end{bmatrix}, I' = \begin{bmatrix} 0 \\ 8 \end{bmatrix}, U' = \begin{bmatrix} \emptyset \\ 1 \end{bmatrix}, Z' = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{4.2}$$

CSP

The following matrices summarize the Strategies of the robot and are necessary to solve the CSP problem :

- $V_{S\times K}$, the Produced Utility matrix, where v_s is the produced utility of Strategy s for class k. Like Desires, Strategies can only be associated with one Desire class each, so each row of the V matrix contain a single non-zero item.

- $L_{S\times K}$: the Strategy Dependency matrix. Rows of the L matrix can contain multiple non-zero items, as a Strategy is allowed to have l_s in multiple classes.

- $\Phi_{S\times 1}$, the vector of subsets of message filters configured by Strategy s.

For the illustrative example, matrices V and L, where column 1 correspond to **SLAM** and column 2 to **GoTo**, are :

$$V = \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix}, L = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \tag{4.3}$$

To solve the CSP problem, Desire class fulfillment vector $F_{K\times 1}$, where $f_k \in \{0, 1\}$ represents the fulfilment of Desire class k, is introduced as internal variables to be used for the optimization process. Based on (P', I', U', Z') and the Strategy matrices, a set of constraints, named $\Gamma = \{\gamma_1, \gamma_2, \gamma_3, ...\}$, is derived according to the following conditions :

- For $u'_k > 0$, which indicates a Desired Utility associated to the Desire class k to be fulfilled (when $f_k = 1$), γ_1 is introduced as in (4.4) to ensure that the Desired Utilities are met for the associated Desire class k :

$$\gamma_1 = \left\{ \sum_s a_s(v_{s,k} - l_{s,k}) \geq f_k u'_k \middle| 1 \leq k \leq K, u'_k > 0 \right\} \tag{4.4}$$

- For $u'_k = 0$, the Desired Utility indicates that fulfilling the Desire class k and its associated Strategy dependencies l_s should be prevented, as expressed by (4.5) :

$$\gamma_2 = \left\{ \sum_s a_s v_{s,k} = 0, \sum_s a_s l_{s,k} = 0 \middle| 1 \leq k \leq K, u'_k \leq 0 \right\} \tag{4.5}$$

- For undefined values of u'_k, γ_3 avoids fulfilling Desires that are not implicitly or explicitly declared by Motivations :

$$\gamma_3 = \left\{ \sum_s a_s(v_{s,k} - l_{s,k}) \geq 0, f_k = 0 \middle| 1 \leq k \leq K, u'_k = \emptyset \right\} \tag{4.6}$$

- For security-related Desires, fulfillment constraints are added for their associated Desire classes :

$$\gamma_4 = \{f_k = 1 | 1 \leq k \leq K, z'_k = 1\} \tag{4.7}$$

For the illustrative example, three constraints are generated : one from γ_1 with $k = 2$, and the pair from γ_3 with $k = 1$:

$$\begin{aligned}
\Gamma &= \left\{ \sum_s a_s \left(v_{s,2} - l_{s,2}\right) \geq f_2 u'_2, \sum_s a_s \left(v_{s,1} - l_{s,1}\right) \geq 0, f_1 = 0 \right\} \\
&= \{a_1(0-0) + a_2(8-0) \geq f_2 \cdot 1, a_1(8-0) + a_2(0-1) \geq 0, f_1 = 0\} \\
&= \{8a_2 \geq f_2, 8a_1 - a_2 \geq 0, f_1 = 0\}
\end{aligned} \tag{4.8}$$

Using Γ, search for all possible solutions of A and F is performed. Table 4.1 illustrates the search space of A and F for the illustrative example. The values for $I \cdot F$ and $\sum_k \sum_s f_k a_s v_{s,k}$ are shown separately for Ω. Of all 16 possible combinations of A and F, only four (marked by * in A^T) satisfy all constraints in Γ.

When only considering constraints in Γ, multiple solutions for A exist, as seen in Tab. 4.1 with $A \in \{[0,0]^T, [1,0]^T, [1,1]^T\}$. The objective is to find the solution A' that maximizes the optimization function Ω, expressed by :

$$\Omega = I \cdot F + \sum_k \sum_s f_k a_s v_{s,k} \tag{4.9}$$

where the left part of the summation aims to maximize the intensity of the Desires to fulfill, and the right part aims to maximize the Utility of the Desires to fulfill. If more than one pair of vectors A and F maximize Ω, the first one to be found is selected. In the illustrative example, the only solution is $A = [1,1]^T$, since for $A = [0,1]^T$, **GoTo** is fulfilled but $(8a_1 - a_2) \ngeq 0$.

Intention Generation

Tableau 4.1 Search space for the illustrative example.

A^T	F^T	Γ	Ω
$[0,0]^*$	$[0,0]$	$\{0 \geq 0, 0 \geq 0, 0 = 0\}$	$0+0=0$
$[0,0]$	$[0,1]$	$\{0 \ngeq 1, 0 \geq 0, 0 = 0\}$	$8+0=8$
$[0,0]$	$[1,0]$	$\{0 \geq 0, 0 \geq 0, 1 \neq 0\}$	$0+0=0$
$[0,0]$	$[1,1]$	$\{0 \ngeq 1, 0 \geq 0, 1 \neq 0\}$	$8+0=8$
$[0,1]$	$[0,0]$	$\{8 \geq 0, -1 \ngeq 0, 0 = 0\}$	$0+0=0$
$[0,1]$	$[0,1]$	$\{8 \geq 1, -1 \ngeq 0, 0 = 0\}$	$8+0=8$
$[0,1]$	$[1,0]$	$\{8 \geq 0, -1 \ngeq 0, 1 \neq 0\}$	$0+8=8$
$[0,1]$	$[1,1]$	$\{8 \geq 1, -1 \ngeq 0, 1 \neq 0\}$	$8+8=16$
$[1,0]^*$	$[0,0]$	$\{0 \geq 0, 0 \geq 0, 0 = 0\}$	$0+0=0$
$[1,0]$	$[0,1]$	$\{0 \ngeq 1, 8 \geq 0, 0 = 0\}$	$0+0=0$
$[1,0]$	$[1,0]$	$\{0 \geq 0, 8 \geq 0, 1 \neq 0\}$	$0+8=8$
$[1,0]$	$[1,1]$	$\{0 \ngeq 1, 8 \geq 0, 1 \neq 0\}$	$0+8=8$
$[1,1]^*$	$[0,0]$	$\{8 \geq 0, 7 \geq 0, 0 = 0\}$	$0+0=0$
$[1,1]^*$	$[0,1]$	$\{8 \geq 1, 7 \geq 1, 0 = 0\}$	$8+8=16$
$[1,1]$	$[1,0]$	$\{8 \geq 0, 7 \geq 0, 1 \neq 0\}$	$0+0=0$
$[1,1]$	$[1,1]$	$\{8 \geq 1, 7 \geq 1, 1 \neq 0\}$	$8+16=24$

The Intentions of the robot are derived using the selected Strategies in A' to define the set of filter values Φ' and the set of non-null configuration parameters P'. In the illustrative example, Strategies for both **SLAM** and **GoTo** are selected, setting all $\phi_{x-SLAM2D}$ and $\phi_{x-Plan\&FollowPath-y}$ to 1, and $p'_{Plan\&FollowPath} = \{0, 10, 0\}$ to make the robot move 10 m to the left.

4.3 Perceptual Filtering in HBBA

Processing all the Perception, the Behavior and the Action Selection and Control modules simultaneously overloads IRL/TR's computational capacity. For instance, the OpenNI-based Kinect capture module cannot produce frames at more than 12 Hz (15 Hz is required), and the Plan & Follow Path module cannot maintain its control loop period of 100 ms to properly navigate the environment. Deactivating Perception modules would reduce computational load but with the cost of loosing perceptual capabilities, and therefore the robot would not be able to accomplish the intended task. Perceptual filtering, i.e., adapting the rate at which the stimuli reach the processing modules, is an alternative in which computational load is reduced without loosing specific capabilities. In HBBA, this can be achieved by using message filters with ϕ values ranging from 1 to ∞ (rather than being 1 or ∞, an all-or-nothing policy). To understand such influences on computational load, we

identified two categories of modules that could benefit from perceptual filtering : modules with linear complexity, and modules with non-linear complexity.

4.3.1 Modules with Linear Complexity

A module with linear complexity is a module whose CPU time consumption is proportional to the amount of input data it receives. While processing time might slightly change depending on the input data, the overall complexity remains $O(N)$. For instance, a module such as Face Detector consumes CPU time and produces face detection events proportionally to the rate of its inputs. Of course, as the detection event frequency decreases, the Utility of the module decreases too : processing images only once every 10 sec would severely limit tracking and people recognition.

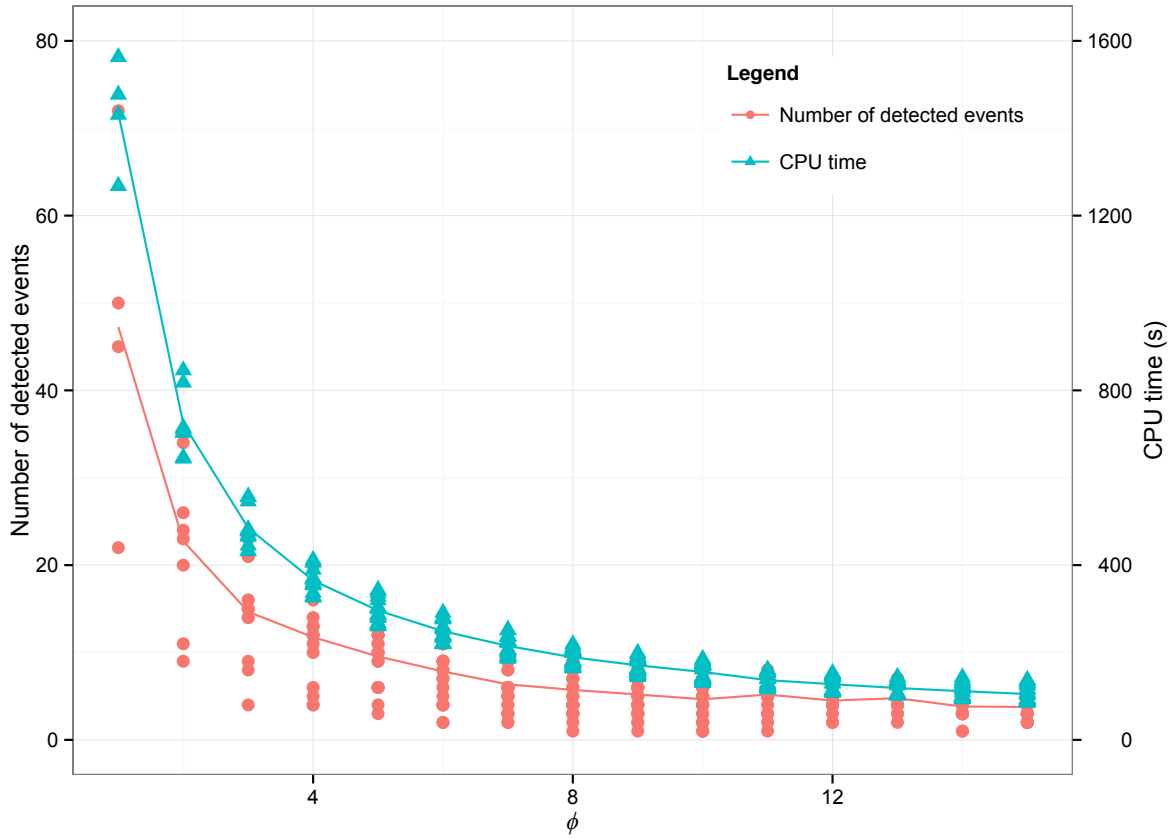


Figure 4.4 Combined number of detected events (left, in red) and CPU time consumed (right, in green) for both QR-Code Detector and Face Detector in relation to ϕ. The lines connect the average sums of detected events and of CPU time.

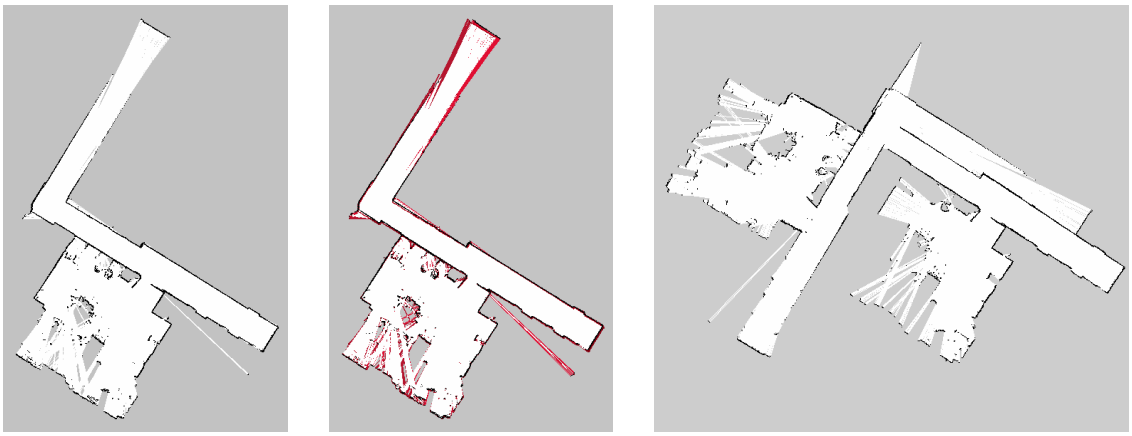
To illustrate cases of modules with linear complexity, $\phi_{x-QR-CodeDetector}$ and $\phi_{x-FaceDetector}$ were modulated from 2 to 15 (for the IMV camera) and 2 to 10 (for the Kinect), to keep the minimum input data frequency at 1 Hz. To present every possible evenly-spaced image sequences to these Perception modules, different starting offset values were tested for each test case. For instance, $\phi = 4$ means testing four image sequences of frame indices : $[1, 5, 9, 13...]$, $[2, 6, 10, 14, ...]$, $[3, 7, 11, 15, ...]$, and $[4, 8, 12, 16, ...]$. Figure 4.4 illustrates their impact on CPU load and the sum of detected events (i.e., QR-codes and faces). As expected, the distributions for both the CPU time and the number of detected events decrease with the amount of data received, following a $\frac{1}{\phi}$ relation.

Loud Noise Detector also has linear complexity, but typically consumes less than one percent of CPU time on IRL-1/TR, and thus its impact is negligible and can be ignored. Furthermore, while Voice Localizer also has linear complexity, it presents a special case where its input signal cannot be decimated, as this module expects a continuous stream of audio data from multiple microphones. In the presented scenario, a single audio stream message contains an audio frame of 512 samples for each microphone. With a sampling frequency of 48 kHz, this corresponds to approximately 10.6 ms of audio data. Simply dividing audio stream data on a 10.6 ms frame basis has undesirable effects on the integrity of the audio signal, and makes the stream unusable by the ManyEars algorithm.

4.3.2 Modules with Non-Linear Complexity

The CPU time consumed by a module may also depends on the content of its input data, and not only on the amount of data it receives. For instance, a robot in motion requires more CPU time for the SLAM 2D module compared to when it is immobile, as its input laser range data change and initiate mapping of new regions. While acquiring laser range finder data at 40 Hz is useful for detecting obstacles rapidly, it produces excessively redundant data, especially for a robot moving at low speed. Conversely, decreasing the input frequency can make map construction and localization impossible.

To determine the impact of the laser range finder data frequency on map production, the SLAM 2D module was tested with $\phi_{x-SLAM2D}$ ranging from 1 to 400, lowering the original 40 Hz down to 0.1 Hz. The CPU time consumed by the SLAM 2D module was monitored over a 3 minutes sequence of laser range finder data and odometry of the robot in motion (maximum velocity : 0.45 m/s). Maps were extracted for each $\phi_{x-SLAM2D}$ tested and compared with the map generated at 40 Hz.



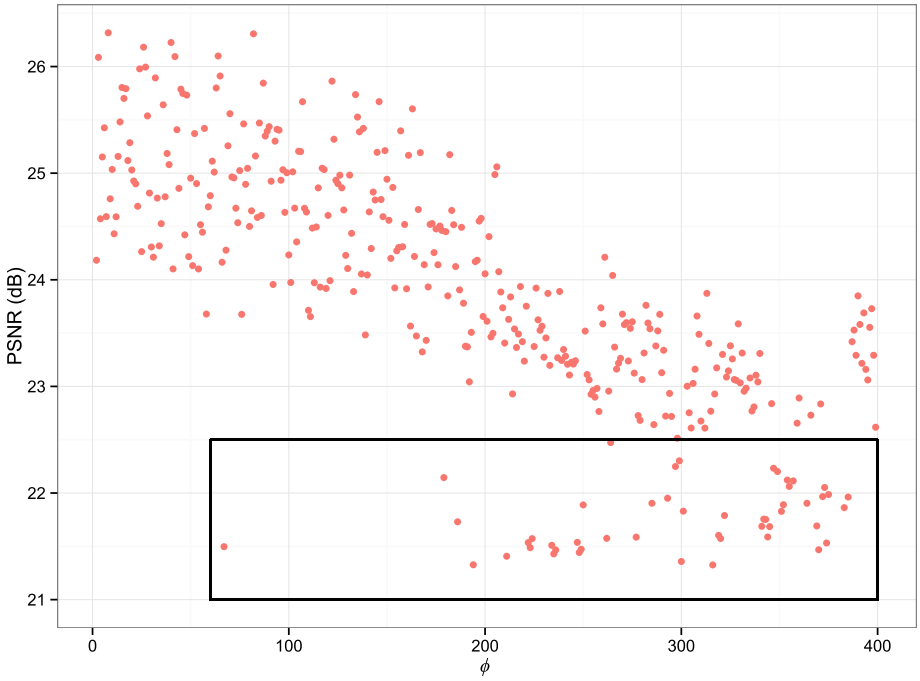
(a) 40 Hz ($\phi = 1$) (b) 1 Hz ($\phi = 40$) (c) 0.215 Hz ($\phi = 186$)

Figure 4.5 Maps of the same area generated by SLAM 2D in relation to $\phi_{x-SLAM2D}$.

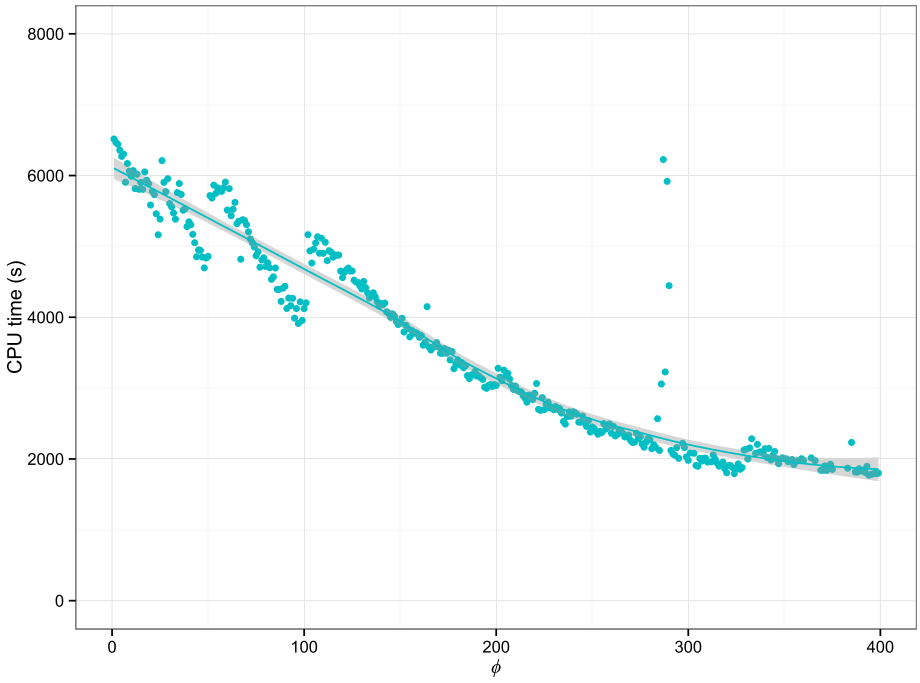
Figure 4.5 presents three examples of maps generated at 40 Hz, 1 Hz and 0.215 Hz. The map generated at 1 Hz, while showing slight differences (identified in red) on the edges of walls, is sufficiently accurate. However, the map at 0.215 Hz has important flaws, with duplicated hallways and rooms : the room at the top left of the map should be the same as the one in the bottom right. Also, the L-shaped corridor is in a wrong orientation and has two adjacent corridors.

Figure 4.6(a) shows the Peak Signal-to-Noise Ratio (PSNR) of maps for $\phi_{x-SLAM2D}$ ranging from 1 to 400. The reference used to compute the PSNR values was the map generated without filtering ($\phi_{x-SLAM2D} = 1$). Outliers in the PSNR values have been used to quickly identify major flaws out of the 400 maps generated. The first occurrence of such flaws happens at $\phi_{x-SLAM2D} = 67$, with a PSNR value of 21.5 dB. Furthermore, they started to appear more frequently with higher divider values, starting at $\phi_{x-SLAM2D} = 186$.

Figure 4.6(b) illustrates the CPU time consumed by the SLAM 2D module for $\phi_{x-SLAM2D}$ ranging from 1 to 400. As expected, CPU time consumption is of non-linear complexity in relation to ϕ. The sawtooth pattern below $\phi_{x-SLAM2D} = 150$ may be related to the odometry data frequency, which is fixed at approximately 100 Hz. As laser data messages are time-filtered inside the SLAM 2D module to be properly transformed in the fixed odometry frame of the robot, some frequencies generate aliasing between the two data sources, causing variable amount of messages to be discarded before they are considered.



(a) PSNR



(b) CPU time

Figure 4.6 PSNR and CPU time for SLAM 2D in relation to $\phi_{x-SLAM2D}$. The black rectangle for PSNR identifies measurements where major flaws started to appear, and the line for CPU time was computed using polynomial local regression.

4.3.3 Perceptual Filtering in HBBA

A perceptual filtering mechanism can be added to HBBA as follows.

- Define strategies with $\phi_s = \{\phi_a, \phi_b, \phi_c, ...\}, \phi \in [1, \infty]$, allowing ϕ to take on values between 1 and ∞.

- Define criteria c related to Strategy s for constraint optimization. These criteria, indexed by j, are :

  1. CPU load $c_1 \in [0, m_1]$, with $m_1 = 100$ the maximum CPU load.

  2. Mutual-exclusivity $c_j \in \{0, m_j\}, j = 2, 3, ...$, with $m_j = 1$, to identify strategies that cannot be selected simultaneously. Distinct mutual-exclusivity criteria are added for each set of mutually-exclusive strategies, such as strategies that are associated to the same Desire class but using different ϕ. By doing so, each of these additional criteria allows the CSP algorithm to select only one strategy in the set of possible strategies.

- Define a fifth set of constraints to add to Γ :

$$\gamma_5 = \left\{ \sum_s a_s c_{s,j} \leq m_j \middle| 1 \leq j \leq J \right\} \tag{4.10}$$

  If all the constraints in Γ cannot be met, Strategies that consume less resources have to be selected, or fulfillment of Desires have to be dropped, based on the optimization function Ω defined by (4.9). Ω attempts to maximize $I \cdot F$ so that fulfilment of intense Desire classes are preferred, and to favour Strategies with high Produced Utility value while still allowing to select Strategies with lower Produced Utility satisfying CPU load and mutual-exclusivity.

As with matrices U, R and Φ, costs for Strategies are gathered in the criteria matrix $C_{S \times J}$, and the platform capabilities in the vector $M_{J \times 1}$, where J is the number of resources to manage.

To continue with the illustrative example, two mutually-exclusive Strategies related to **LocateFaces** are added, along with the CPU load cost for each strategy, as expressed by Table 4.2 and matrices (4.11). CPU costs (c_1) where obtained by measuring the average CPU load (in %) for each Strategy in the same conditions as in Sec. 4.3. Thus, maximum CPU load (m_1) was set to 100.

Tableau 4.2 Strategies for the illustrative example.

s	k	ϕ	v	l_1	c_1	c_2
1	**SLAM**	40	8	0	20	0
2	**GoTo**	1	8	1	30	0
3	**LocateFaces**	1	8	0	60	1
4	**LocateFaces**	2	4	0	30	1

$$V = \begin{bmatrix} 8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 8 \\ 0 & 0 & 4 \end{bmatrix}, L = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, C = \begin{bmatrix} 20 & 0 \\ 30 & 0 \\ 60 & 1 \\ 30 & 1 \end{bmatrix}, M = \begin{bmatrix} 100 \\ 1 \end{bmatrix} \tag{4.11}$$

Adding a Desire instance for **LocateFaces** with $i = 8$ results in following matrices :

$$P = \begin{bmatrix} \{\} & \{10,0,0\} & \{\} \\ \{\} & \{0,10,0\} & \{\} \\ \{\} & \{\} & \{\} \end{bmatrix}, I = \begin{bmatrix} 0 & 4 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 8 \end{bmatrix}, U = \begin{bmatrix} \emptyset & 1 & \emptyset \\ \emptyset & 1 & \emptyset \\ \emptyset & \emptyset & 1 \end{bmatrix}, Z = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.12}$$

$$P' = \begin{bmatrix} \{\} \\ \{0,10,0\} \\ \{\} \end{bmatrix}, I' = \begin{bmatrix} 0 \\ 8 \\ 8 \end{bmatrix}, U' = \begin{bmatrix} \emptyset \\ 1 \\ 1 \end{bmatrix}, Z' = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.13}$$

Γ has the following constraints :

$$\gamma_1 = \left\{ \begin{array}{c} 8a_2 \geq f_2 \\ 8a_3 + 4a_4 \geq f_3 \end{array} \right\} \tag{4.14}$$

$$\gamma_3 = \left\{ \begin{array}{c} 8a_1 - 8a_2 \geq 0 \\ f_1 = 0 \end{array} \right\} \tag{4.15}$$

$$\gamma_5 = \left\{ \begin{array}{c} 20a_1 + 30a_2 + 60a_3 + 30a_4 \leq 100 \\ a_3 + a_4 \leq 1 \end{array} \right\} \tag{4.16}$$

The two parts of the next constraints are available in Eq. 4.16. The first part limits the total CPU load to 100, and the second part avoids activating mutually exclusive Strategies 3 and 4. Table 4.3 lists a subset of solutions (the valid solutions are marked by * in A^T) fulfilling at least one Desire class, with their associated Ω values. Solution $A = [1, 1, 0, 1]^T$ maximizes

Ω out of the valid solutions. The solution $A = [1, 1, 1, 0]$, while fulfilling all Desires, does not satisfy constraints in γ_5 by consuming too much CPU ($\sum_s a_s c_1 = 110 \not\leq m_1 = 100$).

Tableau 4.3 Solutions for the illustrative example with **LocateFaces**.

A^T	F^T	γ_5	Ω
$[1, 1, 0, 0]^*$	$[0, 1, 0]$	$50 \leq 100, 0 \leq 1$	$8 + 16 = 24$
$[0, 0, 1, 0]^*$	$[0, 0, 1]$	$60 \leq 100, 1 \leq 1$	$8 + 8 = 16$
$[0, 0, 0, 1]^*$	$[0, 0, 1]$	$30 \leq 100, 1 \leq 1$	$8 + 4 = 12$
$[1, 1, 0, 1]^*$	$[0, 1, 1]$	$80 \leq 100, 1 \leq 1$	$16 + 20 = 36$
$[1, 1, 1, 0]$	$[0, 1, 1]$	$110 \not\leq 100, 1 \leq 1$	$16 + 24 = 40$
$[1, 1, 1, 1]$	$[0, 1, 1]$	$140 \not\leq 100, 2 \not\leq 1$	$16 + 28 = 44$

4.4 Implementation and Results

ROS offers a message passing structure well-suited for perceptual filtering. By adding an intermediary node between Sensor and Perception modules to implement message filters, messages can be dynamically discarded to prevent data from reaching a module (also interfaced using a ROS node) without having to alter its original code, thus enabling easier reuse of processing modules. These intermediary nodes act as time-decimating frequency message filters, and are implemented using a modified version of the *throttle* Utility found in the *topic_tools* package, allowing the frequency divider ϕ to be modified by the Intention Translator module for each filter at runtime. The CSP implementation is based on Google's Operations Research Tools[3]. The Intention Translator also uses Google V8[4]-based Javascript virtual machine to generate the Intentions of the robot. Each Strategy has a script that performs ROS service calls and topic publications to set message filters Φ' and parameters P'. The implementation of the Intention Translator module, message filters and related code is publicly available online[5].

Table 4.4 lists the Strategies developed for the tour-guide configuration task. Average CPU load was measured individually for each strategy with $\phi = 1$, except for **SLAM**, where a single Strategy with $\phi = 40$ was used. For **FindQRCodes** and **LocateFaces**, additional Strategies were created based on observations made in Section 4.3.1, reducing CPU costs proportionally as ϕ rises. Accordingly, Produced Utility v for **FindQRCodes** and **LocateFaces** Strategies were determined proportionally to the amount of input data,

[3]http ://code.google.com/p/or-tools/
[4]http ://code.google.com/p/v8/
[5]http ://github.com/francoisferland/hbba/

Tableau 4.4 Strategies available in the tour-guide configuration task.

s	k	ϕ	v	l	c_1	c_2	c_3
1	**FindQRCodes**	1	8	-	80	1	0
2	**FindQRCodes**	2	4	-	40	1	0
3	**FindQRCodes**	4	2	-	20	1	0
4	**FindQRCodes**	8	1	-	10	1	0
5	**LocateFaces**	1	8	-	60	0	1
6	**LocateFaces**	2	4	-	30	0	1
7	**LocateFaces**	4	2	-	15	0	1
8	**LocateFaces**	8	1	-	8	0	1
9	**DetectNoises**	1	8	-	0	0	0
10	**LocateVoices**	1	8	-	60	0	0
11	**SLAM**	40	8	-	20	0	0
12	**Teleoperation**	1	8	-	0	0	0
13	**GoTo**	1	8	**SLAM** : 1	30	0	0

assuming that producing twice the amount of detected events makes these modules twice as useful.

Two experiments were conducted using these strategies : a controlled experiment during which Desire instances were specifically generated to validate the Strategy selection process with perceptual filtering, and an experiment with IRL-1/TR with and without using perceptual filtering.

4.4.1 Validation of Strategy Selection with Perceptual Filtering

Figure 4.7 represents the timeline of events in this experiment. The relative message rate is $\frac{\phi_{max}}{\phi}$, where ϕ_{max} is the maximum value of ϕ for Strategies associated with the Desire class. The timeline of events was generated as follows :

- Event 1 : a Desire instance of **FindQRCodes** is created, with $u = 1$, a low Desired Utility. Since there was sufficient CPU available, Strategy 1 is selected ($A' = 1$), with $\phi = 1$.

- Event 2 : a Desire instance of **LocateFaces** is added. This has the effect of replacing Strategy 1 by Strategy 2 and of adding Strategy 5 ($A' = \{2, 5\}$), reducing the throughput of images going to QR-Code Detector ($\phi = 2$) to satisfy the CPU load criterion. At this point, since both Desires have identical i and u values, along with similar CPU costs for their related Strategies, reducing the throughput of Face Detector would also have been a valid solution, as both solutions maximize Ω equally.

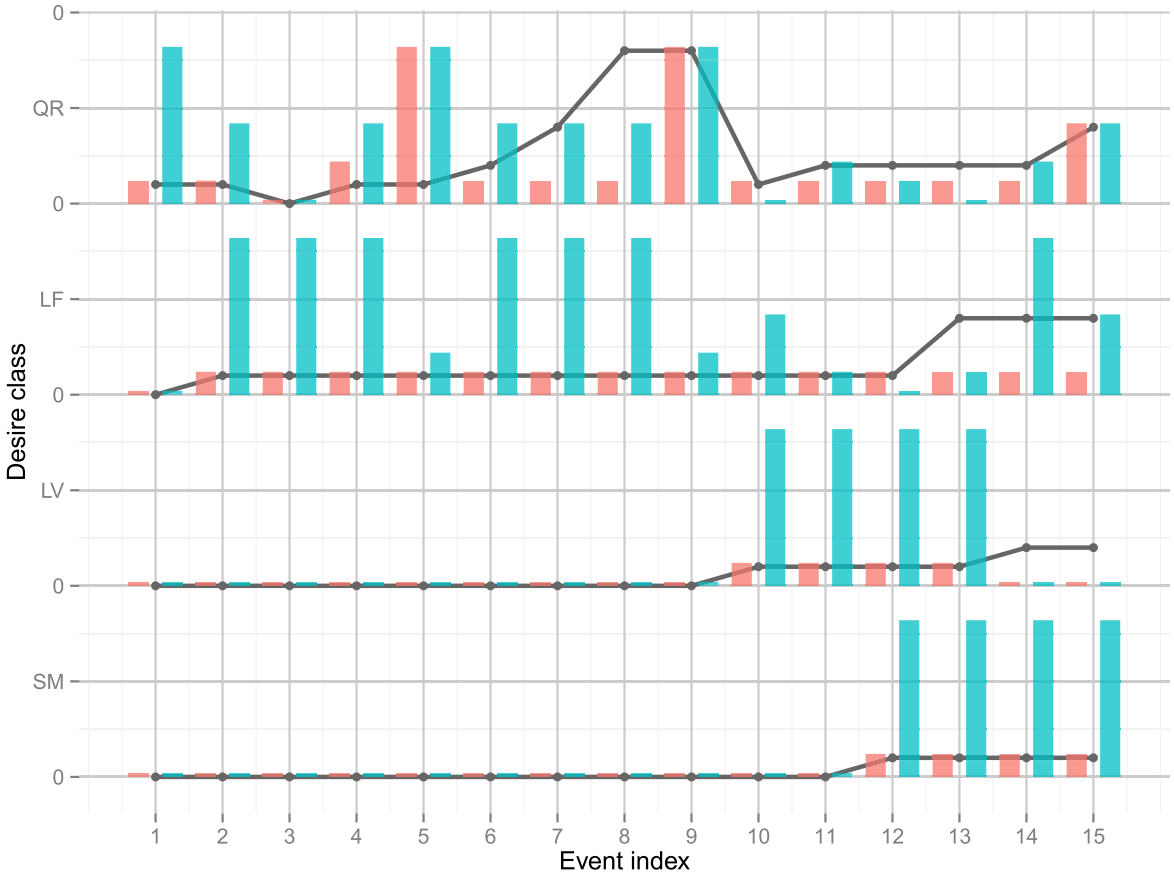


Figure 4.7 Timeline of events in the Intention Translator for four Desire classes : **FindQRCodes** (QR), **LocateFaces** (LF), **LocateVoices** (LV), **SLAM** (SM), with a scale of 0 to 10 (5 being the middle line). Intensity i'_k are represented as dark lines and points, Desired Utility u'_k as red bars, and blue bars represent the relative message rate, scaled from 0 to 8.

- Event 3 : the **FindQRCodes** instance is removed ($A' = \{5\}$), thus completely stopping data going to QR-Code Detector ($\phi = \infty$).

- Event 4 : a **FindQRCodes** instance is added, but with a higher Desired Utility ($u = 2$). To satisfy the γ_2 constraints from (4.4), Strategy 2 with $v = 4$ is selected ($A' = \{2, 5\}$) with $\phi = 2$.

- Event 5 : the Desired Utility of the **FindQRCodes** instance is increased to 8. Only Strategy 1 satisfies this constraint, with $\phi = 1$ (full throughput). This results in decreasing the throughput of Face Detector ($\phi = 4$), selecting Strategy 7 instead of Strategy 5 ($A' = \{1, 7\}$).

- Events 6 to 8 : these events illustrate how high Intensity values are independent of v. Lowering the Desired Utility of **FindQRCodes** back to $u = 1$, the Intentions stay the same as the Intensity of the **FindQRCodes** instance increases from $i = 2$ to $i = 8$. From the point of view of (4.9), having $\phi = 1$ with both **LocateFaces** and **FindQRCodes** produce the same value for $\sum_k \sum_s f_k a_s v_{s,k}$, or the total Produced Utility for Desires to fulfill, in this case both of them ($F = \{1, 1\}$). Thus, both solutions are valid, and the first one found is selected ($A' = \{2, 5\}$).

- Event 9 : u for **FindQRCodes** is increased, resulting in the same Intentions as in Event 5 ($A' = \{1, 7\}$).

- Event 10 : a Desire instance for class **LocateVoices** is introduced, along with a return to previous requirements for **FindQRCodes**($u = 1$). In this situation, fulfilling all three Desires can be achieved by selecting Strategies 4, 10, 11, resulting with $\Omega = 13$. However, two solutions exists with $\Omega = 14$: $A' = \{2, 10\}$ and $A' = \{6, 10\}$. Both solutions ignore one Desire, but are seen more valuable because of their higher Utility production, even if the $S \cdot F$ part Ω is lower. The selection process results in dropping **FindQRCodes** ($A' = \{6, 10\}$), but dropping **LocateFaces** would have been an equivalent choice.

- Event 11 : i for the **FindQRCodes** instance is increased, making it more desirable than the **LocateFaces** instance. This creates a new solution with $\Omega = 14$, which results in exchanging Strategy 6 for 8 and adding Strategy 3 ($A' = \{3, 8, 10\}$).

- Event 12 : an instance for **SLAM** is added, decreasing the throughput of QR-Code Detector to the minimum ($\phi = 8$) to free enough resources for both **SLAM** and **LocateVoices** instances. However, the **LocateFaces** instance must be dropped ($A' = \{4, 10, 11\}$).

- Event 13 : the opposite of Event 11 happens, and **LocateFaces** is preferred to **FindQRCodes**($A' = \{8, 10, 11\}$).

- Event 14 : a second **LocateVoices** instance is introduced, but with higher i than the one already present, and with $u = 0$ to inhibit this Desire class. This prevents Strategy 10 from being selected, as imposed by constraints in γ_2 given by (4.5). This results in the selection of Strategies 3 and 5 for **FindQRCodes** and **LocateFaces** ($A' = \{3, 5, 11\}$), as more CPU time becomes available by deactivating Voice Localizer.

- Event 15 : i and u of the **FindQRCodes** instance is increased so that v becomes equal for both **FindQRCodes** and **LocateFaces**, resulting in the selection of Strategies 2 and 6 ($A' = \{2, 6, 11\}$).

4.4.2 Validation on IRL-1/TR

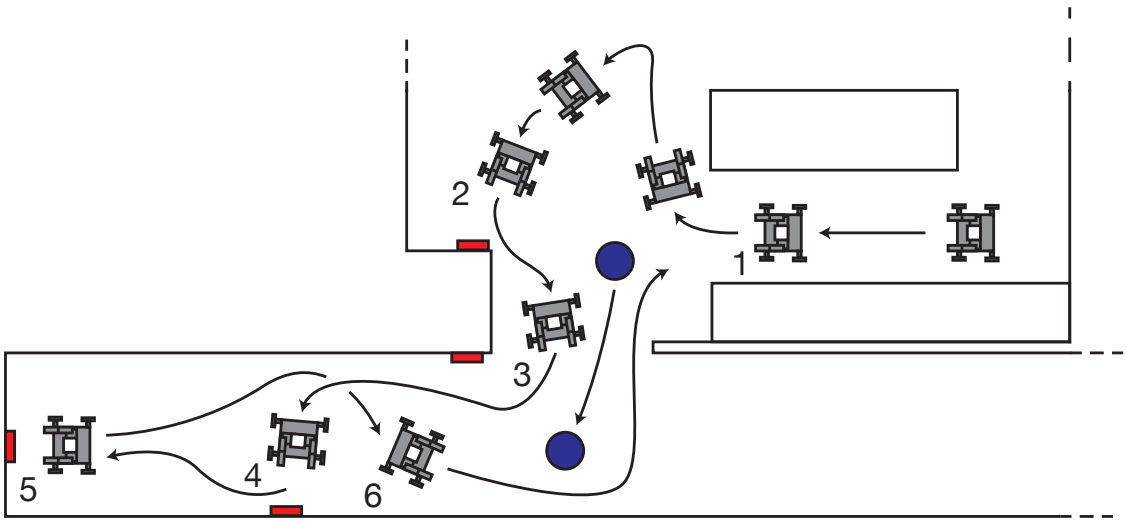


Figure 4.8 The path followed by IRL-1/TR in the tour-guide configuration task. The robot is shown in gray. The QR-Code locations are indicated by the red rectangles. The blue circles represent the two locations occupied by the operator. The numbers refer to events in the scenario.

Table 4.5 lists the Desire instances produced by *Scenario Coordinator* during the experiment conducted with IRL-1/TR. The **GoTo** instance is parametrized with the starting location of the robot, i.e., $(x, y, \theta) = (0, 0, 0)$. The Strategies available for this scenario are the same as in the example given in Sec. 4.4.1, and are shown in Tab. 4.4. The experiment was performed in an open environment. IRL-1/TR was teleoperated through a path, shown in Fig. 4.8, with Desire instances for **FindQRCodes**, **LocateFaces**, **DetectNoises**, **SLAM** and **Teleoperation**, while instances for **LocateVoices** and **GoTo** were created or destroyed according to the conditions outlined in Section 4.2.4. During that path, three QR-codes were meant to be encountered, the operator posing the robot in front of QR-codes for a duration of approximately 2 sec (events 2, 4 and 5). At the third QR-code (event 5), the operator instructed IRL-1/TR to autonomously return to its starting position by pressing Gamepad button 3. A final QR-code was passed by IRL-1/TR (event 6) on its way back, testing the ability of QR-Code Detector to perceive QR-codes in its peripheral vision using the IMV camera. The operator positioned himself in front of

IRL-1/TR before the first QR-code (event 1). He then moved to another location in the robot's path, and was again in the robot's field of view before the second QR-code, and then he attracted the robot's attention by clapping his hands (event 3). The same interaction was repeated when IRL-1/TR initiated going back to the starting location (event 6). Additionally, people walked next to the robot and external noises such as doors being closed occurred randomly. The path was repeated ten times without and with perceptual filtering. Without perceptual filtering, ϕ was set to 1 for all message filters.

Tableau 4.5 Desire instances produced by *Scenario Coordinator*.

d	k	p	i	u	z
1	**FindQRCodes**	-	8	1	0
2	**LocateFaces**	-	8	1	0
3	**DetectNoises**	-	8	1	0
4	**LocateVoices**	-	8	1	0
5	**SLAM**	-	8	1	0
6	**Teleoperation**	-	8	1	1
7	**GoTo**	$(0, 0, 0)$	8	1	0

Figure 4.9 shows the average global CPU load on IRL-1 main computer for the 20 trials of the experiment, generated using Linux `mpstat` Utility. With Hyper-Treading, the CPU load is calculated with eight virtual cores instead of the four physical ones, underestimating the real CPU load. Hyper-Threading exploits idle moments in the processor pipeline to allow an additional thread to run simultaneously. This is presented to the operating system as an additional, virtual CPU core for each physical one. However, as these idle moments depend on the type of load occurring on the CPU, for instance if branch predictions mistakes occur frequently or not. Thus, the performance of these additional virtual cores cannot be compared to the one of the physical core, and a 100 % total CPU load is rarely observable. As expected, variance analysis of the data suggests that the CPU load is significantly lower with perceptual filtering compared to without ($N = 2916$, $p < 2 \times 10^{-16}$). A slight raise in the CPU load with perceptual filtering is visible around 100 sec, which corresponds to the introduction of the **GoTo** Desire instance. Figure 4.10 and Fig. 4.11 present the sum of QR-codes and faces detected every second over the ten trials without and with perceptual filtering. These results are analyzed in the following subsections.

Without Perceptual Filtering

While detection of QR-codes and faces occurred more frequently without perceptual filtering than with perceptual filtering, signs of CPU overload were observed :

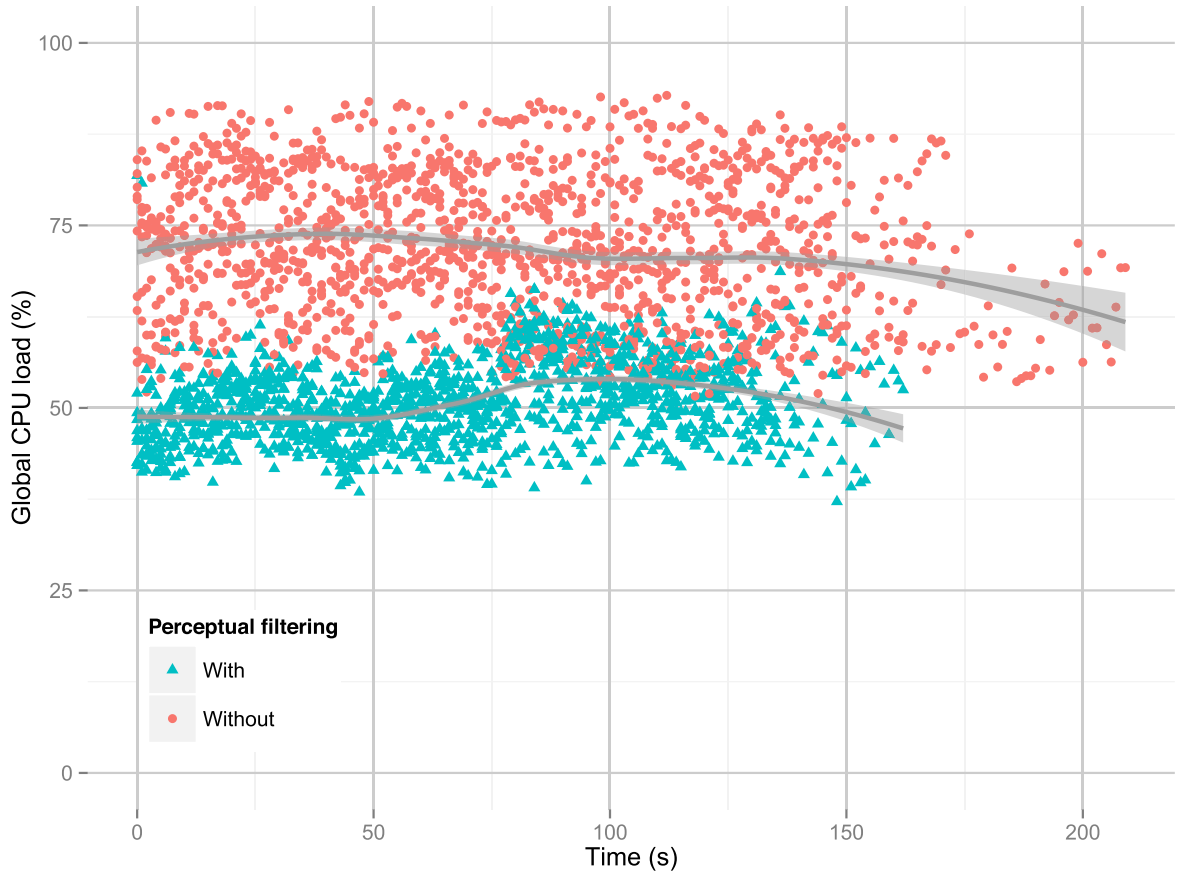


Figure 4.9 Global CPU load for the tour-guide configuration tasks, without and with perceptual filtering.

- In every trial without perceptual filtering, Plan & Follow Path momentarily failed to process data from the Laser Range Finder and the Kinect at 10 Hz. In seven trials, this was severe enough to result in an erratic, stuttering movement of the mobile base for part or all of the autonomous navigation stage of the experiment, as velocity control was stopped because of insufficient laser data. This explains why some trials without perceptual filtering are longer than all those with perceptual filtering, as shown above 160 sec in Fig. 4.9 to 4.11. IRL-1/TR completely stopped near the end of three of those trials, and the operator had to teleoperate the robot back to its starting point.

- In six trials, the OpenNI-based Kinect capture module failed, as it could not properly process its input data with the CPU time allowed. Such failure has potentially dangerous consequences : instead of stopping the production of image frames, it sends partially or completely zero-ed data for both RGB color and depth. Instead of detecting a sensor failure, Plan & Follow Path is led to believe that there are no

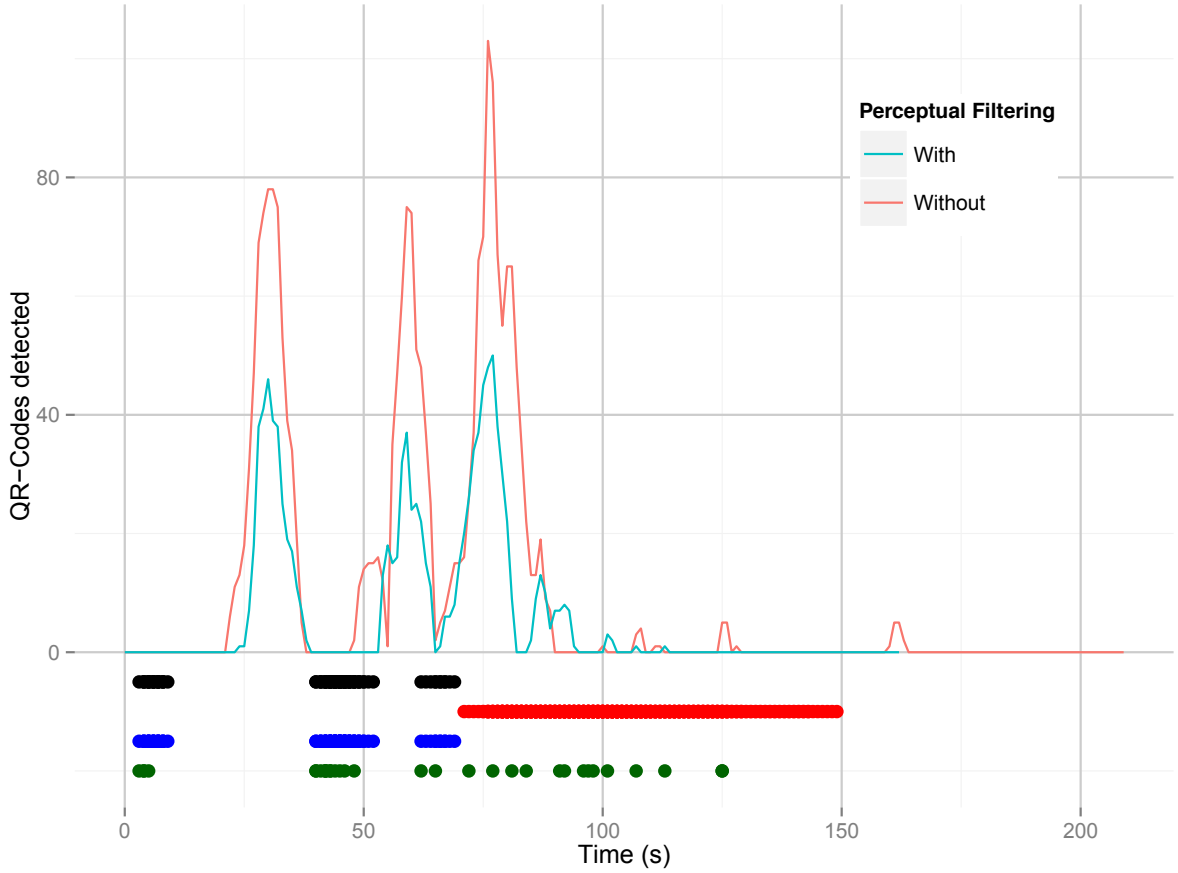


Figure 4.10 QR-Codes detected for all trials in both conditions. The black bars show when Strategy $s = 2$ was removed from the selection, making ϕ go from 2 to ∞. The red bar shows when the Strategy for **GoTo** ($s = 13$) was selected, and the blue bar when the Strategy for **LocateVoices** ($s = 10$) was selected. The green dots represent when loud noises were detected.

obstacles in the field of view of the Kinect, which could have resulted in collisions that would normally be avoided.

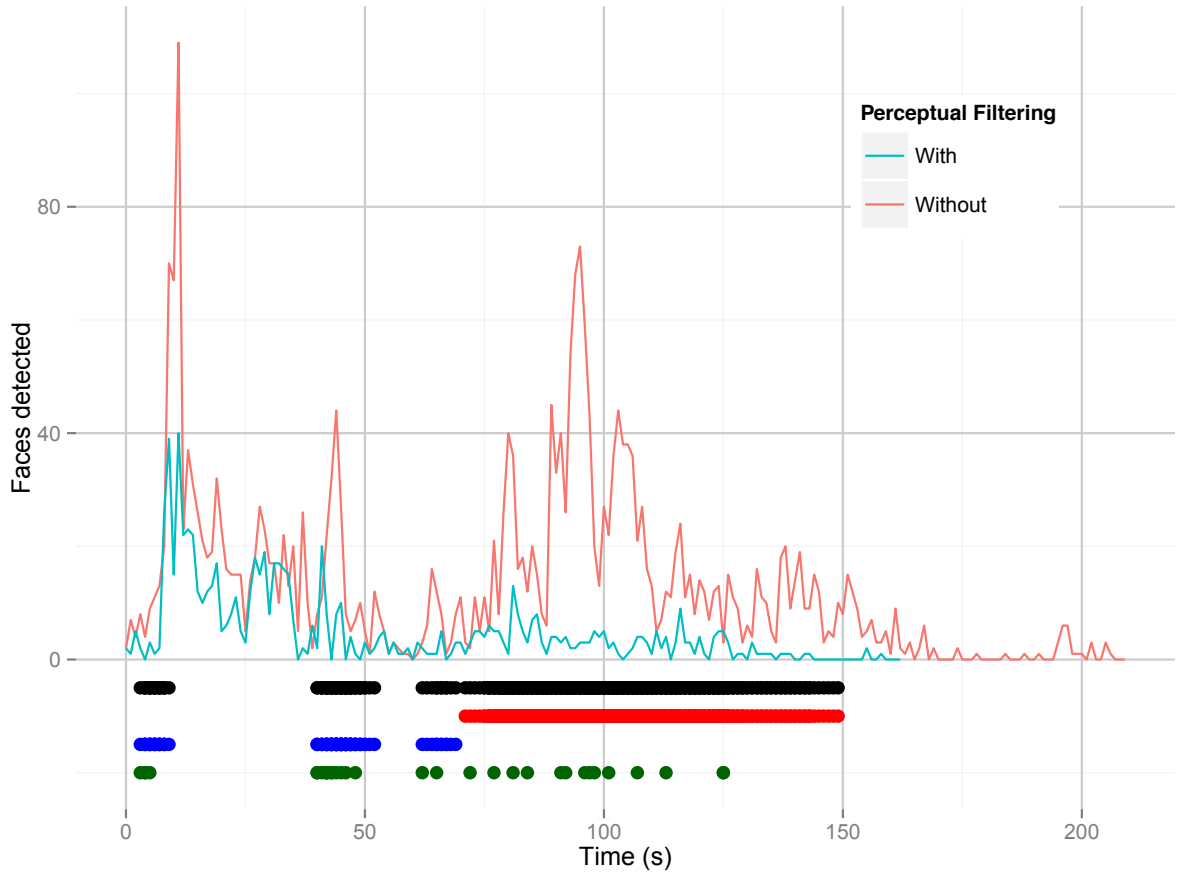


Figure 4.11 Face detected for all trials in both conditions. The black bars show when Strategy $s = 6$ was replaced by $s = 8$, making ϕ go from 2 to 8. The red bar shows when the Strategy for **GoTo** ($s = 13$) was selected, and the blue bar shows when the Strategy for **LocateVoices** ($s = 10$) was selected. The green dots represent when loud noises were detected.

With Perceptual Filtering

With perceptual filtering, IRL-1/TR did not experience crashes of the OpenNI-based Kinect capture module or erratic control from Plan & Follow Path. Figure 4.12 illustrates i'_k for five Desire classes, along with the resulting message passing rate $\frac{8}{\phi}$ as modulated by perceptual filtering, in a typical trial. Noise events, triggering a rise of Intensity for the **LocateVoices** class, are clearly seen, and result in changing A' from $\{2, 6, 9, 11, 12\}$ to $\{8, 9, 10, 11, 12\}$, dropping the fulfilment of **FindQRCodes**. The first noise events detected at the beginning of a path are false detections corresponding to the initialization period of Loud Noise Detector (as indicated in Sect. 4.2). The other two noise events were created when the operator clapped his hands. Introducing a **GoTo** instance also created a similar drop in **LocateFaces** fulfillment, and CPU availability was insufficient to allow fulfillment of the **LocateVoices** instance when the third noise event occurred.

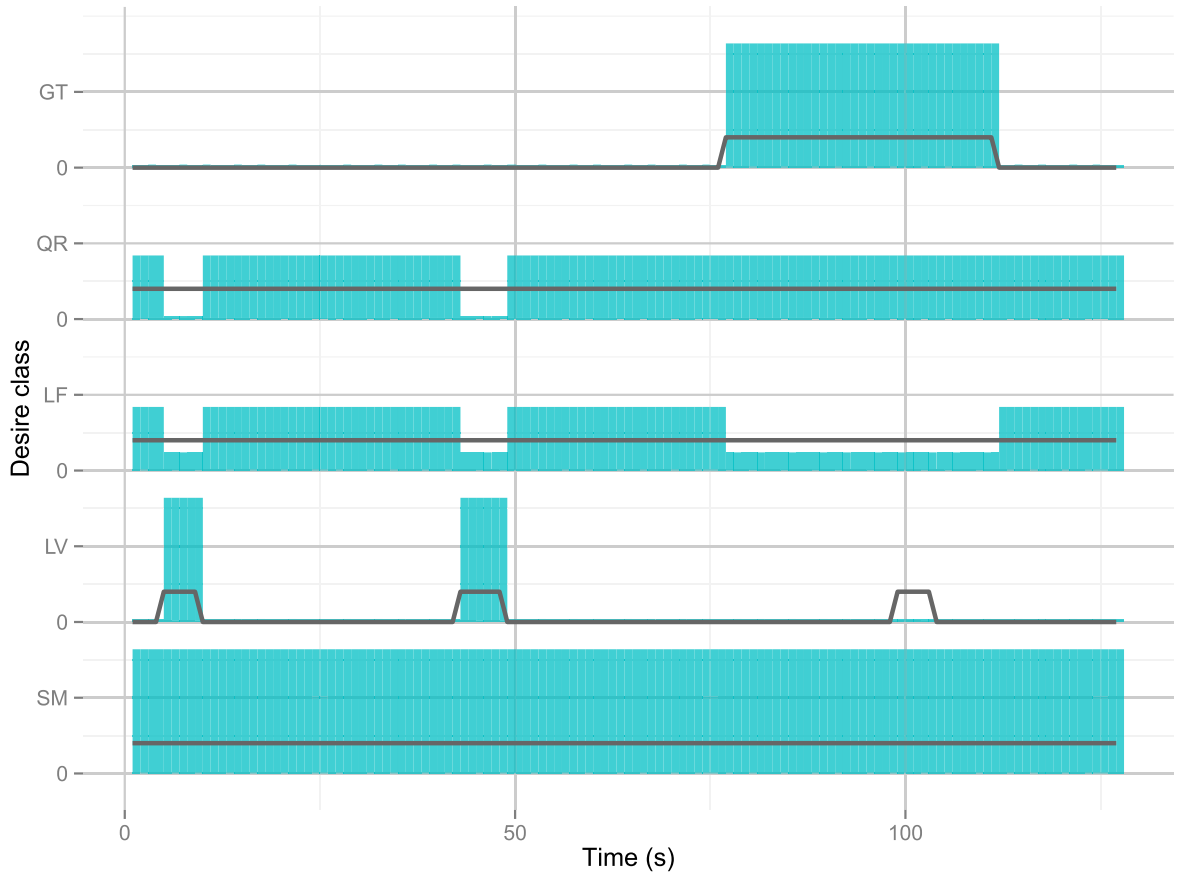


Figure 4.12 Timeline of Desires and Intentions of a typical trial with perceptual filtering. The black lines represents i'_k (scaled down to 25 % for readability reasons) for each class : **GoTo** (GT), **FindQRCodes** (QR), **LocateFaces** (LF), **LocateVoices** (LV), **SLAM** (SM). The blue bars represent the relative message rate, scaled from 0 to 8.

Regarding QR-codes, detections of the first three QR-codes were successful in all 10 trials, but detection of the fourth QR-code was less frequent, as seen by the small peaks passed 100 sec in Fig. 4.10. This is because the fourth QR-code was perceived while IRL-1/TR was in movement, and less frames per second reaching QR-Code Detector implies a lower probability of obtaining at least one clear image of the QR-code. Furthermore, the effect of ignoring the **FindQRCodes** instance when a noise event occurred can be observed by the detection delay around 50 sec : as the first images of the second QR-code often arrived shortly after the first hand clap, they were occasionally ignored in the condition with perceptual filtering.

Finally, similar results are observed with faces : face detections occurred often in the first 50 sec of trials with perceptual filtering, which corresponds to the second encounter with the operator, after the detection of the first QR-code. This is sufficient to trigger a change in the Intentions of IRL-1/TR, which selects a Strategy for **LocateVoices**. However, face detections occurred less often when the robot autonomously returned to its starting position, because Strategy 8 set $\phi = 8$, the lowest rate possible : the selection of Strategies for both **GoTo** and **SLAM** did not leave enough resources to be able to select a Strategy for **LocateFaces**. A simple change in the scenario, where the Intensity of **LocateFaces** would be set higher relatively than the one for **FindQRCodes**, could have triggered a Strategy selection switch between **LocateFaces** and **FindQRCodes**, as demonstrated with events 10 and 11 in Sec. 4.4.1.

4.5 Discussion

The use of our perceptual filtering mechanism requires defining appropriate levels of i, u and v. This can be managed by following simple design guidelines :

- u must be clearly documented for each class. In the tour-guide configuration task, Desired Utility values for instances of **FindQRCodes** and **LocateFaces** are implicitly related to the maximum expected count of detected QR-codes and faces, respectively.

- i and u values must be normalized between classes for the optimization function Ω to work.

Also, it could have been possible to remove the Intensity-Based Selection step and let the CSP handle the selection of the Desire instances that best satisfy the Ω optimization. For instance, this would permit the fulfilment of less-intensive Desire instances of the same

Desire class if they have lower u. However, we believe that having an easily predictable mechanism for resolving conflicts between multiples instances of a single Desire classes simplifies the process of designing Motivation modules. This predictability becomes especially important when Desire instances have to inhibit specific behaviors : as a Desire instance with $u = 0$ prevents the fulfillment of a Desire class, an instance of the same class with lower i but with $u > 0$ could be fulfilled instead, as its v might be higher in the right part of equation (4.9) defining Ω than the fulfillment of an instance with higher i in the right part. The Intensity-Based Selection step makes it possible to prevent that from happening. It also highlight the need for separate Utility and Intensity parameters. In most cases, it can be seen as desirable to produce high Utility for Desires with high Intensity. However, inhibition of Desires would not be possible if both values were represented by the same parameter. As preventing a Desire class from being fulfilled does not produce Utility, it would not be considered by the second half of Ω, and a different scheme had to be designed.

Perceptual filtering may not be applied to all processing modules. For instance, for **LocateVoices**, it is impossible to decimate the input signal of Voice Localizer, as explained in Sec. 4.3. Instead, multiple instances of **LocateVoices** can be created, each with a different configuration such as the use of two, four, and eight microphones, with and without sound source separation, to provide modules with various levels of computational requirements. Message filters with all-or-nothing Strategies can then be used. The same technique can be applied to Behavior modules with heavy computational requirements, such as Plan & Follow Path, by adding for instance a simpler behavior module that only leads the robot in the general direction of the target while performing local obstacle detection, providing rudimentary but often sufficient navigation capabilities over short distances.

Finally, Strategies can be designed to direct sensor data to identical Perception modules running on different computers. For modules that do not maintain an internal state (e.g., Face Detector, unlike SLAM 2D), no synchronisation is necessary between the various instances running on different computers, and thus makes the transition transparent. The network bandwidth required could be used as an additional criterion c to be considered by the Intention Translator.

4.6 Conclusion

With more and more capabilities being integrated into the same mobile robot, this paper presents how a robot control architecture can benefit from using a mechanism to adapt the

rate at which stimuli reach the processing modules. Perceptual filtering makes it possible to dynamically adjust the computational load of CPU on-board a robot based on its Desires and the situations experienced in its operating environment. The resulting filtering mechanism can be associated to a unified model of selective attention, integrating both Early Selection and Late Selection theories [Lavie, 1995, 2005]. It also provides an elegant way to reuse code of processing modules across projects, since integration only influences what their input and output ports are connected to.

In future work, we plan to continue exploring the capabilities provided by our perceptual filtering mechanism, by working on the following :

- Definition of a mathematical relation between CPU cost c_1, v and ϕ for Strategies. For instance, with **FindQRCodes** and **LocateFaces**, intervals linking ϕ and v to c_1 could be defined, instead of having to define discrete Strategies for all possible values of ϕ.

- Modulation of Desire Intensity i_d to provoke priority changes in Desire fulfilment based on which Behavior modules effectively control the robot. In HBBA, this is referred to as behavior exploitation [Michaud *et al.*, 2010b]. Desires that are actively fulfilled can be detected by monitoring exploitation of Behavior modules that produce Utility in their class. Unfulfilled or ignored Desires can also be detected in the same fashion. Coupled with an artificial emotions module, an emotion such as anger could be generated in relation to these ignored Desires, which could then be translated into a rise in Intensity for unfulfilled Desires. This would result in a rise of priority for these Desires and, eventually, better chances of being selected at the Intensity-Based Selection and CSP stages.

- Evaluation of different definitions of Ω. As a first extension, weights could added to each part of the sum to modify the balance between Intensity and Utility requirements. Furthermore, resource consumption could be added to Ω. For instance, adding a term such as $(m_1 - \sum_s a_s c_{s1})$ would integrate CPU usage in the strategy selection process, prioritizing solutions that consume less power, which could be useful in situations where battery capacity is at its lowest.

Finally, we are currently validating the use of our selective attention mechanism in different applications (i.e., tour-guide, telepresence, fetch-and-deliver task) and robot platforms, to demonstrate the versatility of the overall framework.

Acknowledgements

The authors want to thank Marie Avril, Ronan Chauvin, Francis Leconte, Dominic Létourneau and Adrien Tronche for their help in implementing and validating various subsystems on IRL-1.

CHAPITRE 5

Conclusion

La contribution scientifique principale de la présente thèse est de proposer un mécanisme intégré d'attention sélective pour l'activation de modules de traitement avec la capacité de filtrage perceptuel des stimuli. L'utilité d'une telle capacité est démontrée avec un vrai robot humanoïde interactif, en exploitant une architecture comportementale de contrôle en robotique. Ainsi, la thèse propose un mécanisme pour mettre en œuvre une approche hybride d'attention sélective, combinant les principes de la théorie de la sélection hâtive et de la sélection tardive tels que présentés à la section 2.1.

La flexibilité et la généricité du mécanisme d'attention sélective ont aussi pu être démontrées par son utilisation dans plusieurs projets connexes, tels que :

- L'intégration de l'algorithme de cartographie et localisation simultanées RTAB-SLAM [Labbé et Michaud, 2013] aux stratégies pour la classe de désirs **SLAM**, ce qui permet à des modules de motivation d'exploiter ce module sans modification à la définition de ses désirs.

- L'ajout à IRL-1 d'une technique de guidage par contact physique utilisant les bras en impédance articulaire variable. Le guidage du robot peut être effectué en étirant ou en compressant ses bras, tout en laissant libre l'inclinaison autour de l'épaule, ce qui permet de poser les bras à la hauteur préférée par l'utilisateur, et de les ranger le long du torse du robot lorsque le guidage n'est pas souhaité. L'approche a été validée avec 15 volontaires [Ferland *et al.*, 2013] et est depuis ce temps intégrée à plusieurs démonstrations et expériences en tant que comportement pour HBBA.

- La conception d'une interface de téléopération intégrant deux représentations visuelles novatrices : la mesure des forces d'interaction avec l'environnement et la localisation de sources sonores [Reveleau *et al.*, 2014]. La validation de l'approche a été effectuée avec 31 volontaires, et elle suggère que ces représentations sont utiles et appropriées pour des opérateurs à distance d'un robot humanoïde.

- Le développement d'une mémoire épisodique influencée par un modèle d'émotions artificielles [Leconte *et al.*, 2014]. Des émotions, comme la colère lorsque le robot n'arrive pas à accomplir une tâche, vient influencer le taux de mémorisation d'épisodes dans un réseau de neurone à résonance adaptative (*Adaptive Resonance Theory*

[Grossberg, 1976]). Les éléments composants ces épisodes incluent l'exploitation des comportements, et les émotions sont générées à partir des intentions produites par l'*Intention Translator* et l'exploitation (ou non) des comportements liés à ces intentions.

La versatilité de l'architecture résultante se traduit aussi par son exploitation sur différentes plateformes matérielles. En plus d'avoir été validée sur trois variations matérielles d'IRL-1 (IRL-0 dans [Rousseau *et al.*, 2013], IRL-1/AZ3 dans le chapitre 3 et IRL-1/TR dans le chapitre 4), des travaux ont déjà été entamés pour adapter HBBA à des robots Nao d'Aldeberan et l'humanoïde M1 de Meka Robotics à l'ENSTA-ParisTech. Par exemple, un module de détection d'émotions dans le visage développé à cette école a déjà été adapté pour HBBA, et une stratégie qui lui est associée peut maintenant être sélectionnée sur toutes les plateformes ayant une caméra vidéo conventionnelle. Le mécanisme de filtrage perceptif est un élément fondamental pour soutenir cette approche.

Enfin, les possibilités qu'offrent maintenant l'architecture HBBA avec le mécanisme d'attention sélective permet d'aborder l'étude et le développement avec des robots ayant des capacités cognitives plus avancées. Par exemple, il serait intéressant d'évaluer l'interaction entre le filtrage perceptif, la mémoire épisodique et le générateur d'émotions artificielles développés dans le cadre de [Leconte *et al.*, 2014]. Le filtrage influencera nécessairement la quantité d'informations qui parviendront à la mémoire épisodique et, parallèlement à cela, les émotions autant que la mémoire pourront moduler l'intensité des désirs du robot, provoquant potentiellement une sélection différente des stratégies composant les intentions du robot. En fait, cette intégration de la mémoire épisodique ainsi que l'adaptation à l'humanoïde M1 sont à l'origine d'un projet de recherche en cours de développement sur des comportements expressifs adaptatifs pour robots d'assistance [Ferland *et al.*, 2014]. En percevant l'état émotionnel et différentes facettes de la personnalité d'une personne interagissant avec un robot, nous serons en mesure d'ajuster les comportements du robot aux préférences de la personne et ainsi obtenir des interactions plus naturelles et agréables, comme visé dans [Tapus *et al.*, 2008]. Puisqu'une reconnaissance des personnes et une évaluation précise de leurs émotions nécessitent plusieurs modalités de perception visuelles et auditives, l'utilisation d'une approche d'intégration avancée comme celle rendue possible avec HBBA et son mécanisme d'attention sélective va de soi. C'est en étudiant ce genre d'intégration de capacités de perception et d'action évoluées que des robots humanoïdes pourront éventuellement arriver à opérer dans nos environnements de vie au quotidien.

LISTE DES RÉFÉRENCES

Ahonen, T., Hadid, A. et Pietikäinen, M. (2004). Face recognition with local binary patterns. Dans *Computer Vision - ECCV*, Lecture Notes in Computer Science, *volume 3021*. Springer Berlin Heidelberg, p. 469–481.

Albus, J. (1991). Outline for a theory of intelligence. *IEEE Transactions on Systems, Man, and Cybernetics*, volume 21, numéro 3, p. 473–509.

Almàssy, N. (1993). *Bugworld : A Distributed Environment for the Development of Control Architectures in Multi-Agent Worlds* (Rapport technique 93.32). Department of Computer Science, University Zurick-Irchel.

Andronache, V. et Scheutz, M. (2006). ADE - An architecture development environment for virtual and robotic agents. *International Journal on Artificial Intelligence Tools*, volume 15, numéro 2, p. 251.

Antonelli, G., Arrichiello, F. et Chiaverini, S. (2005). The null-space-based behavioral control for mobile robots. Dans *Proceedings of Computational Intelligence in Robotics and Automation*. p. 15–20.

Arbib, M. (1992). Schema theory. Dans Shapiro, S., *Encyclopedia of Artificial Intelligence*, deuxième édition. Wiley, New York, NY, USA, p. 1427–1443.

Arkin, R. C. (1998). *Behavior-Based Robotics*. MIT Press, Cambridge, MA, USA.

Bass, L., Clements, P. et Kazman, R. (2003). *Software Architecture in Practice*. Addison-Wesley Professional, p. 21.

Béchard, M. (2011). *General Social Survey - 2010 Overview of the Time Use of Canadians* (Rapport technique). Statistics Canada, Ottawa, ON.

Borst, C., Wimbrock, T., Schmidt, F., Fuchs, M., Brunner, B., Zacharias, F., Giordano, P. R., Konietschke, R., Sepp, W., Fuchs, S., Rink, C., Albu-Schaffer, A. et Hirzinger, G. (2009). Rollin' Justin – Mobile platform with variable base. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. p. 1597–1598.

Brachman, R. (2006). (AA)AI more than the sum of its parts. *AI Magazine*, volume 27, numéro 4, p. 19–34.

Breazeal, C. et Scassellati, B. (1999). A context-dependent attention system for a social robot. Dans *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 1146–1153.

Breazeal, C. et Scassellati, B. (2000). Infant-like social interactions between a robot and a human caregiver. *Adaptive Behavior*, volume 8, numéro 1, p. 49–74.

Broadbent, D. E. (1958). *Perception and Communication*. Pergamon Press, London, UK.

Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, volume 2, numéro 1, p. 14–23.

Brooks, R. (1990). Elephants don't play chess. *Robotics and Autonomous Systems*, volume 6, p. 3–15.

Brooks, R. (1991). New approaches to robotics. *Science*, volume 253, numéro 5025, p. 1227–1232.

Brooks, R., Breazeal, C., Irie, R., Kemp, C., Marjanovic, M., Scassellati, B. et Williamson, M. (1998). Alternative essences of intelligence. Dans *Proceedings of the National Conference on Artificial Intelligence*. p. 961–968.

Bruyninckx, H. (2001). Open robot control software : the OROCOS project. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. volume 3. p. 2523–2528.

Buerger, S. P. et Hogan, N. (2005). Impedance and interaction control. Dans *Robotics and Automation Handbook*. CRC Press, p. 19.1–19.24.

Bundesen, C. et Habekost, T. (2004). Attention. Dans Lamberts, K., Goldstone, R. L. et Goldstone, R., *Handbook of Cognition*. SAGE Publications, p. 105–129.

Carter, K., Scheutz, M. et Schermerhorn, P. (2009). A humanoid-robotic replica in USARSim for HRI experiments. Dans *IROS Workshop on Robots, Games, and Research*.

Chen, T. et Kemp, C. C. (2011). A direct physical interface for navigation and positioning of a robotic nursing assistant. *Advanced Robotics*, volume 25, p. 605–27.

Chen, T. L. et Kemp, C. C. (2010). Lead me by the hand : Evaluation of a direct physical interface for nursing assistant robots. Dans *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction*. p. 367–374.

Chibani, A., Amirat, Y., Mohammed, S., Matson, E., Hagita, N. et Barreto, M. (2013). Ubiquitous robotics : Recent challenges and future trends. *Robotics and Autonomous Systems*, volume 61, numéro 11, p. 1162 – 1172.

Cooper, A., Reimann, R. et Cronin, D. (2007). *About Face 3 : The Essentials of Interaction Design*. Wiley, Indianapolis, Indiana.

Demiris, Y. et Khadhouri, B. (2006). Hierarchical attentive multiple models for execution and recognition of actions. *Robotics and Autonomous Systems*, volume 54, numéro 5, p. 361 – 369.

Deutsch, J. A. et Deutsch, D. (1963). Attention : Some theoretical considerations. *Psychological Review*, volume 70, numéro 1, p. 80–90.

Deutsch, J. A. et Deutsch, D. (1967). Comments on "Selective attention : Perception or response ?". *The Quarterly Journal of Experimental Psychology*, volume 19, p. 362–363.

Duquette, A., Michaud, F. et Mercier, H. (2008). Exploring the use of a mobile robot as an imitation agent with children with low-functioning autism. *Autonomous Robots, Special Issue on Socially Assistive Robots*, volume 24, numéro 2, p. 147–157.

Endsley, M. R. (1988). Design and evaluation for situation awareness enhancement. Dans *Proceedings of the Human Factors Society 32th Annual Meeting*. p. 97–101.

Eriksen, C. et Hoffman, J. E. (1972). Temporal and spatial characteristics of selective encoding from visual displays. *Perception & Psychophysics*, volume 12, p. 201–204.

Eriksen, C. et Hoffman, J. E. (1973). The extent of processing of noise elements during selective encoding from visual displays. *Perception & Psychophysics*, volume 14, numéro 155-160.

Fauteux, P., Lauria, M., Heintz, B. et Michaud, F. (2010). Dual differential rheological actuator for high performance physical robotic interaction. *IEEE Transactions on Robotics*, volume 26, numéro 4, p. 607–18.

Ferland, F., Aumont, A., Létourneau, D. et Michaud, F. (2013). Taking your robot for a walk : Force-guiding a mobile robot using compliant arms. Dans *Proceedings of the 8th ACM/IEEE International Conference on Human-Robot Interaction*. p. 309–316.

Ferland, F., Clavien, L., Frémy, J., Létourneau, D., Michaud, F. et Lauria, M. (2010). Teleoperation of AZIMUT-3, an omnidirectional non-holonomic platform with steerable wheels. Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 2515 –2516.

Ferland, F., Leconte, F., Tapus, A. et Michaud, F. (2014). An architecture with integrated episodic memory for adaptive robot behavior. Dans *Proceedings of the AAAI Fall Symposium on Artificial Intelligence in Human-Robot Interaction*.

Ferland, F., Létourneau, D., Aumont, A., Frémy, J., Legault, M.-A., Lauria, M. et Michaud, F. (2012). Natural interaction design of a humanoid robot. *Journal of Human-Robot Interaction, Special Issue on HRI Perspectives and Projects from Around the Globe*, volume 1, numéro 2, p. 14–29.

Ferland, F. et Michaud, F. (2014). Perceptual filtering for selective attention in a behavior-based robot control architecture. *Autonomous Robots (soumis en octobre 2014)*.

Ferland, F., Pomerleau, F., Le Dinh, C. et Michaud, F. (2009). Egocentric and exocentric teleoperation interface using real-time, 3D video projection. Dans *Proceedings of the 4th ACM/IEEE International Conference on Human-Robot Interaction*. p. 37–44.

Fox, D., Burgard, W. et Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, volume 4, numéro 1, p. 23–33.

Frémy, J., Michaud, F. et Lauria, M. (2010). Pushing a robot along – A natural interface for human-robot interaction. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, p. 3440–3445.

Gazzaniga, M. S. et Ledoux, J. E. (1978). *The Integrated Mind*. Plenum Press.

Grisetti, G., Stachniss, C. et Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, volume 23, numéro 1, p. 34–46.

Grondin, F., Létourneau, D., Ferland, F., Rousseau, V. et Michaud, F. (2013). The ManyEars open framework. *Autonomous Robots*, volume 34, numéro 3, p. 217–232.

Grossberg, S. (1976). Adaptive pattern classification and universal recoding : II. Feedback, expectation, olfaction, illusions. *Biological Cybernetics*, volume 23, numéro 4, p. 187–202.

Guizzo, E. (2010). When my avatar went to work. *IEEE Spectrum*, volume 47, numéro 9, p. 26–32.

Guizzo, E. (2011). How google's self-driving car works. *IEEE Spectrum Online, 18 octobre*.

Hawes, N., Brenner, M. et Sjöö, K. (2009a). Planning as an architectural control mechanism. Dans *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*. p. 229–230.

Hawes, N., Sloman, A., Wyatt, J., Zillich, M., Jacobsson, H., Kruijff, G., Brenner, M., Berginc, G. et Skocaj, D. (2007). Towards an integrated robot with multiple cognitive functions. Dans *Proceedings of the National Conference on Artificial Intelligence*, Menlo Park, CA ; Cambridge, MA ; London ; AAAI Press ; MIT Press ; 1999. p. 1548.

Hawes, N. et Wyatt, J. (2010). Engineering intelligent information-processing systems with CAST. *Advanced Engineering Informatics*, volume 24, numéro 1, p. 27 – 39.

Hawes, N., Wyatt, J. et Sloman, A. (2009b). Exploring design space for an integrated intelligent system. *Knowledge-Based Systems*, volume 22, numéro 7, p. 509–515.

Hawes, N., Zender, H., Sjöö, K., Brenner, M., Kruijff, G.-J. M. et Jensfelt, P. (2009c). Planning and acting with an integrated sense of space. Dans *Proceedings of the 1st International Workshop on Hybrid Control of Autonomous Systems – Integrating Learning, Deliberation and Reactive Control (HYCAS)*. p. 25–32.

Hirzinger, G., Sporer, N., Schedl, M., Butterfass, J. et Grebenstein, M. (2004). Torque-controlled lightweight arms and articulated hands : Do we reach technological limits now ? *International Journal of Robotics Research*, volume 23, p. 331–340.

Hogan, N. (1985). Impedance control : An approach to manipulation : Part II - Dynamics systems measurement control. *Journal of Dynamic Systems, Measurement, and Control*, volume 107, p. 17.

Jacobsson, H., Hawes, N., Kruijff, G.-J. et Wyatt, J. (2008). Crossmodal content binding in information-processing architectures. Dans *Proceedings of the 3rd ACM/IEEE International Conference on Human-Robot Interaction*. ACM, New York, NY, USA, p. 81–88.

Jain, A. et Kemp, C. C. (2009). Pulling open novel doors and drawers with equilibrium point control. Dans *Proceedings of the IEEE-RAS International Conference on Humanoid Robotics*. IEEE, p. 498–505.

Kelly, A., Chan, N., Herman, H., Huber, D., Meyers, R., Rander, P., Warner, R., Ziglar, J. et Capstick, E. (2011). Real-time photorealistic virtualized reality interface for remote mobile robot control. *International Journal of Robotics Research*, volume 30, numéro 3, p. 384–404.

Kortenkamp, D. et Simmons, R. (2008). Robotic systems architectures and programming. Dans *Handbook of Robotics*, chapitre 8. Springer, p. 187–206.

Labbé, M. et Michaud, F. (2013). Appearance-based loop closure detection in real-time for large-scale and long-term operation. *IEEE Transactions on Robotics*, p. 734–745.

Labonté, D., Boissy, P. et Michaud, F. (2010). Comparative analysis of 3D robot teleoperation interfaces with novice users. *IEEE Transactions on Systems, Man, and Cybernetics-Part B : Cybernetics*, volume 40, numéro 5, p. 1331–1342.

Langley, P., Laird, J. E. et Rogers, S. (2009). Cognitive architectures : Research issues and challenges. *Cognitive Systems Research*, volume 10, numéro 2, p. 141 – 160.

Lauria, M., Legault, M.-A., Lavoie, M.-A. et Michaud, F. (2008). Differential elastic actuator for robotic interaction tasks. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. p. 3606–3611.

Lavie, N. (1995). Perceptual load as a necessary condition for selective attention. *Journal of Experimental Psychology*, volume 21, numéro 3, p. 451–468.

Lavie, N. (2005). Distracted and confused ? : Selective attention under load. *Trends in Cognitive Sciences*, volume 9, numéro 2, p. 75 – 82.

Lavie, N. et Fox, E. (2000). The role of perceptual load in negative priming. *Journal of Experimental Psychology Human Perception and Performance*, volume 26, numéro 3, p. 1038–1052.

Lavie, N. et Tsal, Y. (1994). Perceptual load as a major determinant of the locus of selection in visual attention. *Perception & Psychophysics*, volume 56, numéro 2, p. 183–197.

Leconte, F., Ferland, F. et Michaud, F. (2014). Fusion adaptive resonance theory networks used as episodic memory for an autonomous robot. Dans *Proceedings of the Conference on Artificial General Intelligence*. Lecture Notes in Computer Science, Springer International Publishing, p. 63–72.

Lee, M. K., Forlizzi, J., Rybski, P. E., Crabbe, F., Chung, W., Finkle, J., Claser, E. et Kiesler, S. (2009). The Snackbot : Documenting the design of a robot for long-term human-robot interaction. Dans *Proceedings IEEE/ACM International Conference on Human-Robot Interaction*. p. 7–14.

Legault, M.-A., Lavoie, M.-A., Cabana, F., Jacob-Goudreau, P., Letourneau, D., Michaud, F. et Lauria, M. (2008). Admittance control of a human centered 3 DOF robotic arm

using differential elastic actuators. Dans *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 4143–4144.

Lorenz, K. (1973). *The Foundations of Ethology*. Springer-Verlag, New York, NY, USA.

Mataric, M. et Michaud, F. (2008). Behavior-based robotics. Dans *Handbook of Robotics*, chapitre 38. Springer, p. 893.

Mataric, M. *et al.* (2002). Situated robotics. *Encyclopedia of Cognitive Science*, volume 4, p. 25–30.

Michaud, F. (2002). EMIB – Computational architecture based on emotion and motivation for intentional selection and configuration of behaviour-producing modules. *Cognitive Science Quarterly, Special Issue on "Desires, Goals, Intentions, and Values : Computational Architectures*, volume 3, numéro 4, p. 340–261.

Michaud, F., Boissy, P., Labonté, D., Brière, S., Perreault, K., Corriveau, H., Grant, A., Lauria, M., Cloutier, R., Roux, M.-A., Iannuzzi, D., Royer, M.-P., Ferland, F., Pomerleau, F. et Létourneau, D. (2010a). Exploratory design and evaluation of a homecare teleassistive mobile robotic system. *Mechatronics, Special Issue on Design and Control Methodologies in Telerobotics*, volume 20, numéro 7, p. 751–766.

Michaud, F., Côté, C., Létourneau, D., Brosseau, Y., Valin, J.-M., Beaudry, É., Raievsky, C., Ponchon, A., Moisan, P., Lepage, P., Morin, Y., Gagnon, F., Giguere, P., Roux, M.-A., Caron, S., Frenette, P. et Kabanza, F. (2007a). Spartacus attending the 2005 AAAI conference. *Autonomous Robots, Special Issue on AAAI Mobile Robot Competition*, volume 22, numéro 4, p. 369–384.

Michaud, F., Ferland, F., Létourneau, D., Legault, M. et Lauria, M. (2010b). Toward autonomous, compliant, omnidirectional humanoid robots for natural interaction in real-life settings. *Paladyn Behavioral Robotic Journal*, volume 1, numéro 1, p. 57–65, http ://dx.doi.org/10.2478/s13230–010–0003–3.

Michaud, F., Laplante, J., Larouche, H., Duquette, A., Caron, S., Letourneau, D. et Masson, P. (2005a). Autonomous spherical mobile robotic to study child development. *IEEE Transactions on Systems, Man, and Cybernetics*, volume 35, numéro 4, p. 471–480.

Michaud, F., Letourneau, D., Arsenault, M., Bergeron, Y., Cadrin, R., Gagnon, F., Legault, M.-A., Millette, M., Pare, J.-F., Tremblay, M.-C., Lepage, P., Morin, Y. et Caron, S. (2005b). Multi-modal locomotion robotic platform using leg-track-wheel articulations. *Autonomous Robots, Special Issue on Unconventional Robotic Mobility*, volume 18, numéro 2, p. 137–156.

Michaud, F., Salter, T., Duquette, A. et Laplante, J.-F. (2007b). Perspectives on mobile robots used as tools for pediatric rehabilitation. *Assistive Technologies, Special Issue on Intelligent Systems in Pediatric Rehabilitation*, p. 21–36.

Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., Johnston, D., Klumpp, S., Langer, D., Levandowski, A., Levinson, J., Marcil, J., Orenstein, D., Paefgen, J., Penny, I., Petrovskaya,

A., Pflueger, M., Stanek, G., Stavens, D., Vogt, A. et Thrun, S. (2008). Junior : The Stanford entry in the Urban Challenge. *Journal of Field Robotics*, volume 25, numéro 9, p. 569–597.

Munir, S. et Book, W. (2002). Internet-based teleoperation using wave variables with prediction. *IEEE/ASME Transactions on Mechatronics*, volume 7, numéro 2, p. 124 –133.

Nakadai, K., Okuno, H. G., Nakajima, H., Hasegawa, Y. et Tsujino, H. (2008). An open source software system for robot audition HARK and its evaluation. Dans *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*. p. 561–566.

Neisser, U. (1967). *Cognitive Psychology*. Appleton-Century-Crofts, New York, NY, USA.

Nelson, H. G. et Stolterman, E. (2003). *The Design Way*. Educational Technology Publication, Englewood Cliffs, NJ.

Nielsen, C., Goodrich, M. et Ricks, R. (2007). Ecological interfaces for improving mobile robot teleoperation. *IEEE Transactions on Robotics*, volume 23, numéro 5, p. 927–941.

Nilsson, N. (1969). A mobile automaton : An application of AI techniques. Dans *Proceedings of the First International Joint Conference on Artificial Intelligence*. p. 509–520.

Norman, D. et Shallice, T. (1986). Attention to action : Willed and automatic control of behavior. Dans Davidson, R., Schwartz, G. et Shapiro, D., *Consciousness and Self-Regulation : Advances in Research and Theory*, *volume 4*. Plenum Press, p. 1–17.

Norman, D. A. (1968). Toward a theory of memory and attention. *Psychological Review*, volume 75, p. 522–536.

Ott, C., Eiberger, O., Friedl, W., Bäuml, B., Hillenbr, U., Borst, C., Albu-schäffer, A., Brunner, B., Hirschmüller, H., Kielhöfer, S., Konietschke, R., Wimböck, T., Zacharias, F. et Hirzinger, G. (2006). A humanoid two-arm system for dexterous manipulation. Dans *Proceedings of the IEEE International Conference on Humanoid Robots*. p. 276–283.

Paletta, L., Rome, E. et Buxton, H. (2005). Attention architectures for machine vision and mobile robots. Dans Itti, L., *Neurobiology of Attention*. Elsevier Academic Press, Amsterdam, NL, p. 642–648.

Pfeifer, R. et Scheier, C. (1999). *Understanding Intelligence*. MIT Press, p. 132.

Pirjanian, P. (2000). Multiple objective behavior-based control. *Robotics and Autonomous Systems*, volume 31, numéro 1-2, p. 53–60.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R. et Ng, A. (2009). ROS : An open-source Robot Operating System. Dans *Open-Source Software Workshop of the International Conference on Robotics and Automation*.

Raïevsky, C. et Michaud, F. (2008). Improving situated agents adaptability using interruption theory of emotions. Dans *From Animals to Animats 10 : Proceedings of the 10th International Conference on Simulation of Adaptive Behavior*. p. 301–310.

Rensink, R., O'Regan, J. et Clark, J. (1977). To see or not to see : The need for attention to perceive changes in scenes. *Psychological Science*, volume 8, p. 368–373.

Reveleau, A., Ferland, F., Labbé, M., Létourneau, D. et Michaud, F. (2014). Visual representation of force and sound in an augmented reality teleoperation interface for a mobile robot. *soumis au Journal of Human-Robot Interaction*.

Rich, C., Ponsleur, B., Holroyd, A. et Sidner, C. L. (2010). Recognizing engagement in human-robot interaction. Dans *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. p. 375–382.

Ricks, B., Nielsen, C. et Goodrich, M. (2004). Ecological displays for robot interaction : A new perspective. Dans *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. volume 3. p. 2855 – 2860.

Robinson, D. (2000). *Design and Analysis of Series Elasticity in Closed Loop Actuator Force Control*. PhD thesis, Massachussets Institute of Technology, Cambridge, Boston.

Rousseau, V., Ferland, F., Létourneau, D. et Michaud, F. (2013). Sorry to interrupt, but may I have your attention ? Preliminary design and evaluation of autonomous engagement in HRI. *Journal of Human-Robot Interaction*, volume 2, numéro 3, p. 41–61.

Ruesch, J., Lopes, M., Bernardino, A., Hornstein, J., Santos-Victor, J. et Pfeifer, R. (2008). Multimodal saliency-based bottom-up attention a framework for the humanoid robot icub. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. p. 962–967.

Russel, S. J. et Norvig, P. (2003). *Artificial Intelligence : A Modern Approach*, 2e édition. Pearson Education Inc., Upper Saddle River, New Jersey, USA.

Rusu, R. et Cousins, S. (2011). 3D is here : Point Cloud Library (PCL). Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. p. 1–4.

Salter, T., Michaud, F. et Larouche, H. (2010). How wild is wild ? A taxonomy to characterize the 'wildness' of child-robot interaction. *International Journal of Social Robotics*, volume 2, numéro 4, p. 405–415.

Sasaki, Y., Kaneyoshi, M., Kagami, S., Mizoguchi, H. et Enomoto, T. (2009). Daily sound recognition using pitch-cluster-maps for mobile robot audition. Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 2724–2729.

Scheutz, M. (2006). ADE : Steps toward a distributed development and runtime environment for complex robotic agent architectures. *Applied Artificial Intelligence*, volume 20, numéro 2, p. 275–304.

Scheutz, M. et Andronache, V. (2004). Architectural mechanisms for dynamic changes of behavior selection strategies in behavior-based systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B : Cybernetics*, volume 34, numéro 6, p. 2377–2395.

Scheutz, M., Schermerhorn, P. et Kramer, J. (2006). The utility of affect expression in natural language interactions in joint human-robot tasks. Dans *Proceedings of the 1st ACM SIGCHI/SIGART Conference on Human-Robot Interaction*. p. 226–233.

Scheutz, M., Schermerhorn, P., Kramer, J. et Anderson, D. (2007). First steps toward natural human-like HRI. *Autonomous Robots*, volume 22, numéro 4, p. 411–423.

Scholtz, J., Young, J., Yanco, H. et Drury, J. (2004). Where am I ? Acquiring situation awareness using a remote robot platform. Dans *Proceedings IEEE Conference on Systems, Man and Cybernetics*. p. 2835–2840.

Scholtz, J. C., Antonishek, B. et Young, J. D. (2005). Implementation of a situation awareness assessment tool for evaluation of human-robot interfaces. *IEEE Transactions on Systems, Man, and Cybernetics-Part A : Systems and Humans*, volume 35, numéro 4, p. 450–459.

Shayganfar, M., Rich, C. et Sidner, C. L. (2012). A design methodology for expressing emotion on robot faces. Dans *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 4577–4583.

Tanner, W. P., J. et Swets, J. A. (1954). A decision-making theory of visual detection. *Psychological Review*, volume 61, p. 401–409.

Tapus, A., Ţăpuş, C. et Matarić, M. J. (2008). User-robot personality matching and assistive robot behavior adaptation for post-stroke rehabilitation therapy. *Intelligent Service Robotics*, volume 1, numéro 2, p. 169–183.

Tenorth, M., Klank, U., Pangercic, D. et Beetz, M. (2011). Web-enabled robots – Robots that use the web as an information resource. *Robotics & Automation Magazine*, volume 18, numéro 2, p. 58–68.

Tipper, S. P. (1985). The negative priming effect : Inhibitory effects of ignored primes. *Quarterly Journal of Experimental Psychology*, volume 37A, p. 571–590.

Trafton, J. G., Bugajska, M. D., Fransen, B. R. et Ratwani, R. M. (2008). Integrating vision and audition within a cognitive architecture to track conversations. Dans *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. p. 201–208.

Treisman, A. M. (1960). Contextual cues in selective listening. *Quarterly Journal of Experimental Psychology*, volume 12, p. 242–248.

Treisman, A. M. (1964). Selective attention in man. *British Medical Bulletin*, volume 20, p. 12–16.

Treisman, A. M. (1985). Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, volume 31, numéro 2, p. 156–177.

Treisman, A. M. et Geffen, G. (1967). Selective attention : Perception or response ? *The Quarterly Journal of Experimental Psychology*, volume 19, numéro 1, p. 1–17.

Tsagarakis, N., Metta, G., Sandini, G., Vernon, D., Beira, R., Becchi, F., Righetti, L., Santos-Victor, J., Ijspeert, A., Carrozza, M. *et al.* (2007). iCub : The design and reali-

zation of an open humanoid platform for cognitive and neuroscience research. *Advanced Robotics*, volume 21, numéro 10, p. 1151–1175.

Tsui, K. T. et Yanco, H. A. (2009). Towards establishing clinical credibility for rehabilitation and assistive robots through experimental design. Dans *Proceedings of the Robotics : Science and Systems Workshop on Good Experimental Methodology in Robotics*.

Ude, A., Moren, J. et Cheng, G. (2007). Visual attention and distributed processing of visual information for the control of humanoid robots. Dans *Humanoid Robots : Human-like Machines*. I-Tech, Vienne, AT, p. 423–436.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T. M., Kolski, S., Kelly, A., Likhachev, M., McNaughton, M., Miller, N., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Salesky, B., Seo, Y.-W., Singh, S., Snider, J., Stentz, A., Whittaker, W. R., Wolkowicki, Z., Ziglar, J., Bae, H., Brown, T., Demitrish, D., Litkouhi, B., Nickolaou, J., Sadekar, V., Zhang, W., Struble, J., Taylor, M., Darms, M. et Ferguson, D. (2008). Autonomous driving in urban environments : Boss and the urban challenge. *Journal of Field Robotics*, volume 25, numéro 8, p. 425–466.

Valin, J.-M., Michaud, F. et Rouat, J. (2007a). Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering. *Robotics and Autonomous Systems Journal*, volume 55, p. 216–228.

Valin, J.-M., Yamamoto, S., Rouat, J., Michaud, F., Nakadai, K. et Okuno, H. (2007b). Robust recognition of simultaneous speech by a mobile robot. *IEEE Transactions on Robotics*, volume 23, numéro 4, p. 742–752.

Valli, A. (2008). The design of natural interaction. *Multimedia Tools and Applications*, volume 38, p. 295–305.

Vijayakumar, S., Conradt, J., Shibata, T. et Schaal, S. (2001). Overt visual attention for a humanoid robot. Dans *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. volume 4. p. 2332–2337.

Vrečko, A., Skočaj, D., Hawes, N. et Leonardis, A. (2009). Integration of computer vision components into a multi-modal cognitive system. Dans *Proceedings of the Fourteenth Computer Vision Winter Workshop*. p. 3140–3147.

Weiskrantz, L. (1986). *Blindsight : A Case Study and Implications*. Clarendon Press, Oxford.

Williamson, M. (1995). *Series Elastic Actuators*. Master's thesis, Massachussets Institute of Technology, Cambridge, Boston.

Wolfe, J. (1994). Guided search 2.0. A revised model of visual search. *Psychonomic Bulletin & Review*, volume 1, numéro 2, p. 202–238.

Wyatt, J. L. et Hawes, N. (2008). Multiple workspaces as an architecture for cognition. Dans *Proceedings of the AAAI Fall Symposium on Biologically Inspired Cognitive Architectures*. p. 201–206.

Wyrobek, K., Berger, E., Van der Loos, H. et Salisbury, K. (2008). Towards a personal robotics development platform : Rationale and design of an intrinsically safe personal robot. Dans *Proceedings of the IEEE International Conference on Robotics and Automation*. p. 2165–2170.

Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., Dang, T., Franke, U., Appenrodt, N., Keller, C., Kaus, E., Herrtwich, R., Rabe, C., Pfeiffer, D., Lindner, F., Stein, F., Erbs, F., Enzweiler, M., Knöppel, C., Hipp, J., Haueis, M., Trepte, M., Brenk, C., Tamke, A., Ghanaat, M., Braun, M., Joos, A., Fritz, H., Mock, H., Hein, M. et Zeeb, E. (2014). Making Bertha drive ? An autonomous journey on a historic route. *Intelligent Transportation Systems Magazine, IEEE*, volume 6, numéro 2, p. 8–20.