Architecture robots autonomes : Une architecture bio-inspirée pour réaliser des robots autonomes

Jean Sallantin, Julien Cotret, Eric Bourreau, Emmanuel Peralta

Une architecture bio-inspirée pour réaliser des robots autonomes.

Jean Sallantin* — **Emmanuel Peralta**** — **Eric Bourreau*****

** Laboratoire d'Informatique de Robotique et de Micro électronique de Montpellier, 161 rue Ada, 34392 MONTPELLIER CEDEX 5.*
jean.sallantin@lirmm.fr

*** Université Montpellier II, Place Eugène Bataillon, 34 095 Montpellier Cedex 5*
eperalta@info-ufr.univ-montp2.fr

**** Laboratoire d'Informatique de Robotique et de Micro électronique de Montpellier, 161 rue Ada, 34392 MONTPELLIER CEDEX 5.*
eric.bourreau@lirmm.fr

RÉSUMÉ. *Nous proposons une architecture bio-inspirée fixant un cadre au développement par un robot de son autonomie lors d'un apprentissage interactif. Les modèles psychologiques du développement qui l'inspirent portent sur les procédés de psychisation des animaux et des humains. Ce cadre permet d'aborder de manière expérimentale certaines questions liées à l'incorporation d'un esprit dans une machine. Les choix techniques pris pour le développement du démonstrateur le rendent indépendant du robot choisi. L'illustration est faite avec un robot Tribot piloté via* URBI*, utilisant des logiciels de planification par contrainte développés en* JCHOCO*, un logiciel d'apprentissage de la bibliothèque* WEKA *et un logiciel de contrôle normatif de son comportement* INTEGRE*. Une première expérimentation étudie comment un apprentissage interactif permet à un robot de contrôler un comportement réflexe.*

ABSTRACT. *This article presents a bio-inspired framework allowing a robot to increase its cognitive performances during an interactive learning. This framework is used to test strategies for mind embodiment into a machine.*

MOTS-CLÉS : *apprentissage interactif, robotique autonome, programmation par contraintes, architecture, psychologie*

KEYWORDS: *interactive learning, autonomous robot, constraint satisfaction problem, framework, psychology*

1re soumission à *RSTI - RIA*, le 6 février 2010

Introduction

Cet article présente une architecture bio-inspirée fixant un cadre au développement de son autonomie par un robot. Cette étude aborde ainsi une question à la croisée des recherches des Sciences et Technologies de l'Information et de la Communication (STIC) et des Sciences Humaines et Sociales (SHS). En effet, l'autonomie est une faculté accordée aux être vivants, personnes et institutions mais pas encore aux machines. La faculté d'autonomie n'est pour autant pas définie par cette opposition.

L'architecture logicielle que nous allons décrire contribue à fonder un langage et des concepts unificateurs pour préciser la notion d'autonomie : tant pour des systèmes vivants qu'artificiels. Cette architecture logicielle accorde à une machine une capacité d'apprentissage lui permettant d'atteindre différents degrés d'autonomie lors d'une interaction avec des instructeurs. Elle contribue au fort questionnement passé et actuel sur l'intelligence des machines.

Le développement d'architecture bio-inspirée pour des machines autonomes pose des question de légitimité et de méthode que nous allons signaler succinctement : elle s'affronte à des blocages intellectuels, elle utilise des notions imparfaitement définies, relevant de la construction de la pensée humaine et elle les applique à des machines.

Le premier point est de nature anthropologique et philosophique et il porte sur une réticence à admettre l'intelligence des machines. Des sociologues voient dans cette réticence une influence culturelle de la religion dominante. Alors que dans la culture orientale (Lipoubou, 2005), il est admis que des intelligences simples puissent animer des objets, la culture occidentale demeure réticente à incorporer un esprit dans un objet. La culture occidentale oppose l'immanence d'un corps à la transcendance d'un esprit. Cette opposition se durcit quand on substitue une machine au corps. Des philosophes influents ont pris position : Pascal comme Hobbes défendent que l'on peut ramener la pensée au calcul (Parrochia, 1992).

"*La machine d'arithmétique fait des effets qui approchent plus de la pensée que tout ce que font les animaux ; mais elle ne fait rien qui puisse faire dire qu'elle a de la volonté, comme les animaux.*" (Pascal, Pensées 262).

En second lieu, les méthodes pour aborder des notions imparfaitement définies ont progressé le siècle dernier. Le structuralisme est une démarche formelle prenant sa source dans la linguistique saussurienne qui définit chaque termes par les relations qu'il entretient avec les autres termes. La démarche structuraliste introduit la notion de système et cherche des invariants dans les systèmes sociaux et les systèmes de pensée. Dans les années 70, dans son cours au collège de France sur le Neutre (Barthes, 1977-1978), Roland Barthes a défini la notion d'idéosphère comme la présentation des vues d'un système non dogmatique car inachevé. Pour Barthes, le marxisme, le freudisme sont des exemples d'idéosphères. Une idéosphère est initiée par un discours fondateur sur un système et elle permet d'établir des correspondances avec d'autres discours et de prendre des positions d'adhésion ou de refus. La notion d'autonomie est une

idéosphère car pour spécifier, concevoir et réaliser des systèmes autonomes, les informaticiens ne disposent à ce jour d'aucun formalisme abouti.

Le troisième point aborde celui de nos relations aux travaux récents sur la formalisation du fonctionnement de l'esprit. Selon la démarche structuraliste qui vient d'être présentée, Lacan (Dor, 2002) a étudié le fonctionnement de l'esprit en y intégrant l'influence de l'inconscient. Sa démarche formelle rencontre celle prise en Intelligence Artificielle quand elle ramène la pensée à un langage formel en correspondance avec différentes logiques modales (Cohen *et al.*, 1990). Maintenant de nombreuses études en pédopsychiatrie portent l'interaction précoce entre le bébé et son environnement et examinent les processus de sémiotisation du nourrisson et en particulier l'usage de ses capacités de représentation à fin de communication (Golse, 1999; Golse, 2007). Ces études précisent des démarches expérimentales qui pourraient être reprises dans le cadre de l'acquisition d'une autonomie par une machine.

Le dernier point porte sur la substitution d'une machine au corps. Nous avons maintenant de grandes réalisations de robots menées au Japon et en Corée qui ont promu la vision orientale de la machine intelligente. Ces réalisations permettent des études expérimentales sur les fonctionnalités attribuant des degrés d'autonomie à des machines. Les travaux menés par Oudeyer à l'INRIA (Oudeyer *et al.*, 2007) portent sur une architecture respectant des principes de psychologie du développement et donnant à un robot bébé la possibilité d'être guidé dans sa construction d'un modèle du monde qui accorde ses perceptions aux actions potentielles. A Munich, il est développé un robot qui se déplace dans la ville sans GPS en demandant sa route aux piétons (Bauer *et al.*, 2009).

Présentons des perspectives plus futuristes. Au Japon, Ishii Kayoko s'intéresse à la co-construction de la conscience et de la schizophrénie chez les robots humanoides (Ishii, 2009). Par ailleurs, une réflexion anthropologique (Coeeckelgergh, 2009) renverse même les perspectives et pose les robots comme de quasi humains. La question de caractériser des états d'autonomie et d'intelligence pour des machines en les associant à des capacités de pensée trouve désormais un intérêt autant conceptuel que pratique.

Après avoir présenté différents points de vue sur la question venant des SHS développons le point de vue STIC.

L'autonomie intéressant les roboticiens et les micro-électroniciens est celle qu'il faut donner aux systèmes pour qu'ils puissent acquérir des aptitudes d'interactivité avec d'autres systèmes, hommes et avec l'environnement et qu'ils acquièrent leurs propres critères d'optimisation.

Pour les informaticiens, il s'agit d'une part, de définir formellement une machine autonome par un modèle indépendant de ses réalisations : ce modèle doit définir comment passer d'une réalisation de l'autonomie à une autre et composer des formes d'autonomies. Il s'agit d'autre part, de réaliser des machines autonomes composées de machines autonomes en interaction qui s'adaptent à des environnements changeants. Il faut enfin les rendre capables d'interagir avec des humains.

Le modèle que nous allons défendre se fonde sur la boucle OODA (observe-orient-decide-act) introduite lors de la première guerre du Golfe pour formaliser l'accélération du tempo sur le plan tactique. Il la complète (voir *figure* 3) en y intégrant ce qui nous semble être au cœur des architectures de Brooks (Brooks, 1985) (voir *figure* 1) et des architectures comportementales (voir *figure* 2).

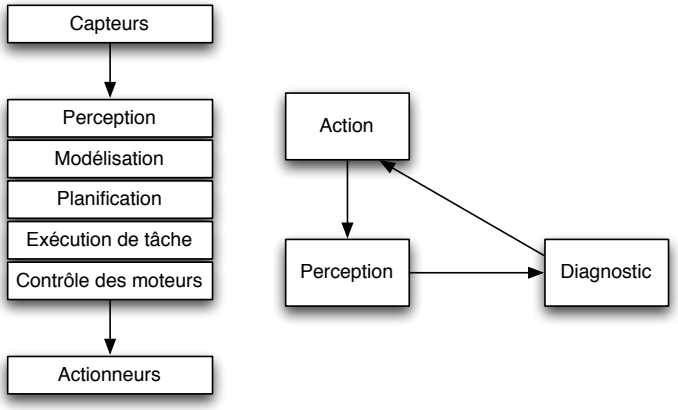

Figure 1. *Au cœur de l'architecture de Brooks, il existe une boucle perception/action non supervisée par un processus de décision.*

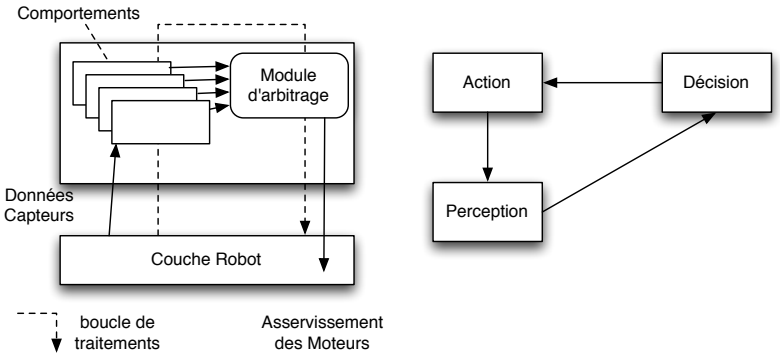

Figure 2. *Au cœur de l'architecture comportementale, il existe une boucle perception/action non réflexe, supervisée par un processus de décision.*

Le plan de cet article est le suivant. Dans une première partie, nous allons décrire et justifier le cadre que nous proposons pour construire l'autonomie d'une machine. Dans une seconde partie, nous allons montrer comment le réaliser pratiquement et nous allons présenter quelques résultats. Dans la dernière partie, nous allons présenter les perspectives expérimentales et formelles ouvertes par ce cadre.

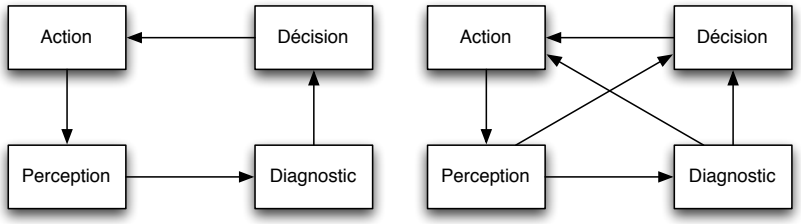

Figure 3. *Présentation de notre modèle complétant la boucle OODA en y intégrant les architectures précédentes.*

1. Le cadre de conception de l'autonomie des machines

Dans cette section, nous allons présenter un vocabulaire permettant de décrire et réaliser des machines présentant différents degrés d'autonomie. Ce vocabulaire va s'établir sur une base structurelle qui a pour fonction d'organiser le traitement de l'information pour tout système ayant un degré d'intelligence. Nous allons introduire ce vocabulaire en nous appuyant sur des exemples impliquant des comportements humains mettant en évidence cette structure fondatrice.

1.1. *Autonomie*

Comme nous l'avons présenté dans l'introduction, nous cherchons à préciser des invariants structurels qui président à la définition d'autonomies pour une machine. Nous allons poser deux principes.

Tout d'abord la notion d'autonomie ne doit pas dépendre de la réalisation matérielle de la machine. Comme l'a envisagé Marvin Minsky dans son livre, la société de l'esprit (Minsky, 1988), une telle machine peut être vue comme une société. Cette vue est développée dans les systèmes multi-agents. On peut imaginer que cette machine ait une taille astronomique, qu'elle soit à notre échelle ou encore qu'elle ait une taille nanoscopique lui permettant d'intervenir dans la matière organique. Donc la notion d'autonomie ne doit pas être affectée par la matérialité de la machine.

En second lieu, une machine autonome bio-inspirée doit avoir la possibilité d'atteindre tous les degrés d'intelligence identifiés en psychologie : stade du miroir, savoir se discerner de son environnement, savoir s'identifier comme un tout, savoir reconnaître son image, savoir s'imaginer, savoir reconnaître les autres, avoir conscience du réel, de ce qui vient, savoir dire "je". Les degrés d'intelligence sont des stades marquant une progression de la production de connaissance, sur les objets, sur soi et sur les autres. L'architecture de la machine doit permettre une telle progression. Les évènements "de vie" de la machine vont motiver cette progression. Les dysfonctionnements

d'une machine intelligente doivent se comprendre comme un mode de fonctionnement "dégradé" de l'architecture. L'architecture est ainsi le cadre qui précise la production d'une intelligence qui, elle, sera marquée par l'histoire.

Notre premier principe est que la notion d'autonomie est indépendante de la nature matérielle d'une machine conçue en respectant l'architecture structurale d'une machine autonome. Notre second principe est que la notion d'intelligence, ses degrés et ses dysfonctionnements doivent s'expliquer en utilisant la description de l'architecture.

1.2. *Introduction au cadre d'élaboration d'une autonomie*

Ayant posé ces principes, nous allons présenter avec des exemples comment les sous-systèmes composant cette architecture sont la condition de l'établissement de l'autonomie.

Notre premier exemple est celui de la conduite accompagnée. Considérons une voiture dans laquelle circulent l'apprenti et son parent l'instruisant. Le parent peut signaler à l'apprenti des événements signifiants (il y a un stop), il peut proférer un diagnostic qu'il infère (tu devrais freiner !), il peut enjoindre de freiner (freine bon sang ! !), mais il ne peut pas pour autant agir sur le frein ou sur le volant. Quatre systèmes sont en relation : le système de l'action, le système de la perception, le système du diagnostic, et le système de la supervision.

L'organisation de ces quatre systèmes est donnée sous la forme d'une boucle. Dans chaque système, sauf celui de l'action, il se passe une interaction entre l'apprenti et l'instructeur. Si nous avions pris pour exemple l'apprentissage de la marche, le parent, en soutenant le bébé, serait intervenu également dans le système de l'action.

Si l'apprenti est trop maladroit, inattentif, sans imagination ou sans discernement alors il sera incapable de réussir son examen de conduite. L'échec est imputé à la défaillance d'un sous système et non à l'histoire de l'apprentissage. Donc la défaillance d'un sous-système légitime la nécessité de la présence de ces quatre sous-systèmes.

Changeons d'exemple et prenons celui de la découverte scientifique. Nous la considérons comme menée par de très nombreux acteurs en interaction. Les actions sont celles des chercheurs, elles produisent des faits signifiants enregistrées comme des données dont on peut examiner la conjonction, la disjonction, la succession. Les diagnostics sont constitués par des articles exprimant des relations générales entre ces signifiants. Ces articles sont soumis à des rapporteurs qui en examinent la validité et suggèrent des révisions.

Dans le premier exemple, l'apprenti constructeur parvient au bout de 3000 Km à conduire de manière autonome car il ne sollicite que rarement et pour des raisons justifiées l'intervention de l'instructeur. Dans le second exemple, le jugement des pairs produit une chaîne de révision conduisant à des publications signifiantes.

Les trois exemples précédents illustrent des systèmes naturellement multi-agents dont la capacité d'intelligence vient de l'interaction entre agents humains. Il s'agit maintenant de voir comment un tel système peut rendre compte de la capacité d'acquisition d'autonomie pour un agent seul en interaction avec d'autres.

Nous allons considérer comme nouvel exemple un système constitué d'un bébé et de la personne qui s'en occupe. Nous allons poser que ce système est semblable à celui constitué de la voiture, l'apprenti et l'instructeur dans la situation de la conduite accompagnée. La différence essentielle et fondamentale vient de ce que la voiture est ici le corps du bébé. Dans cet exemple, l'interaction a pour fonction de permettre le développement des facultés mentales du bébé.

Développons ce dernier exemple. Tout abord, il est avéré que l'interaction stimule un processus d'apprentissage qui perfectionne les systèmes d'Action, de Signification, de Symbolisation et de Communication du bébé. En effet, un bébé dont on ne s'occupe pas a un développement cérébral déficient. Dans le contexte du bébé, on peut distinguer deux phases de développement : celle qui précède le langage et celle qui s'appuie sur ce dernier. Nos principes impliquent qu'il n'y a pas de changement de l'architecture structurant l'activité mentale du bébé lors du passage d'une phase à l'autre.

Cependant l'architecture OODA ne permet pas la prise en compte d'un développement cérébral. Nous allons nous inspirer des recherches en pédopsychiatrie sur le bébé (Golse, 1999) et en psychanalyse (Dor, 2002) pour la compléter. La question de fond largement développée dans ces disciplines est celle de comprendre comment une pulsion de vie illustrée par le besoin de téter du bébé parvient à se transformer en un désir d'interaction, marqué par le désir d'être l'objet de l'attention de la mère. Le bébé parvient à maîtriser une pulsion vitale en lui trouvant un objet de substitution.

Les modèles de fonctionnement de la pensée de Lacan sont inspirés de ceux de Freud et ils s'appuient sur les fondements de la linguistique en introduisant une relation signifiant/signifié telle qu'elle est définie par Saussure (Dor, 2002).

Nous venons d'illustrer comment un système, lors d'interactions, peut avoir la capacité de développer une intelligence comme étant une capacité de penser. Nous allons maintenant présenter des études préliminaires menées au LIRMM sur la mise en pratique du modèle précédent.

L'informatique et la robotique cohabitent au LIRMM. Les roboticiens se préoccupent plus de mécanique et d'automatique. Dans le projet SHERPA[1] il s'agit de fabriquer un bipède et de produire les lois de commande permettant la marche à un robot bipède. Les lois de commande recalculent à un tempo rapide les valeurs à donner aux actuateurs en fonction des valeurs des capteurs.

Le système est ainsi doté d'un contrôle réactif. Les problèmes de contrôle hybride ont été abordés dans une première thèse (Paulin, 2008). Lors de cette, thèse nous avons

1. Site web du département de robotique du LIRMM http ://www.lirmm.fr/∼w3rob/Rob/

vérifié qu'un système à base de contrainte parvient à contrôler l'équilibre d'un robot aussi bien et dans le même tempo que les systèmes classiquement utilisés (Fikes *et al.*, 1971). La thèse de Paulin montre comment conjuguer l'apprentissage d'unités de contrôle élémentaire et la planification d'actions pour réaliser un contrôle réactif. Les unités de contrôle élémentaires sont les éléments du système de perception permettant de construire des plans.

La vidéo présente sur le site de Mathias Paulin[2] montre comment un robot est capable de recalculer de manière compulsive des plans d'action quand son plan est contrarié. Un tel robot n'a aucune capacité de production d'une intelligence quelconque. L'étude, que nous avons menée depuis, est consacrée à la production du cadre structurel en entier de manière à tenter de reproduire les premières étapes de la réalisation d'un esprit lors d'interaction et d'apprentissage.

2. Le projet dubitoide

Le projet "dubitoide" (Bourreau *et al.*, 2009) a pour ambition de produire un système logiciel respectant l'architecture précédente et capable de participer à l'attribution d'une certaine autonomie à une machine. Le nom dubitoide "dubito ergo sum" signale que l'activité cognitive du robot, déployée dans ce cadre conceptuel va pratiquer sur un doute systématique pascalien.

Rappelons que nous avons posé deux principes : l'indépendance de l'architecture à la réalisation matérielle de la machine et l'indépendance de l'architecture aux degrés d'autonomie que la machine va pouvoir atteindre. Un principe informatique s'applique ici : "résoudre des problèmes généraux avec des outils généraux". En effet, deux sous système, Diagnostic et Supervision, utilisent une problématique informatique bien identifiée : la programmation par contraintes (Montanari, 1974; Mackworth, 1977) et celle d'un système expert spécialisé dans le contrôle normatif de comportements[3].

Ces problématiques ont un très large champ d'application. Le système Action est réalisé avec le logiciel URBI de Gostai[4]. Ce logiciel a pour avantage d'être utilisé pour commander de nombreux robots. Il n'est pas pleinement adapté à un contrôle réactif comme celui de l'équilibre du robot. Cependant URBI est fondé sur des principes d'ordonnancement synchrone ou asynchrone de tâches parallèles et séquentielles ; principes qui nous semblent exemplaires.

Le système de perception a été réalisé simplement par une gestion de fichiers enregistrant des séquences d'actions. Le jeu de flèches entre les sous-systèmes exprime des traductions : certains objets d'un premier sous-système seront traduits en objets du second et certaines relations du premier seront traduites en relation du second. Par exemple, les séquences d'actions sont traduites en enregistrement dans le système de perception.

---

2. http ://mathias.paulin.free.fr
3. Intègre : http ://www.normind.com
4. http ://www.gostai.com

La flèche diagonale, qui exprime le contrôle réactif du diagnostic sur les actions, a été spécialement étudiée car elle correspond à une problématique générale de génération de plan. La flèche diagonale, qui exprime l'apprentissage symbolique interactif, a été réalisée avec un souci de généralité. Les données du système de perception sont traduites dans les formats appropriés pour lancer les logiciels de la boîte à outil WEKA (Witten *et al.*, 2005).

2.1. *Expérimentation*

Nous allons reprendre point par point les sous-systèmes du robot. Pour chacun des systèmes, nous décrirons ses objets et les relations entre ces objets. Nous donnerons le système informatique qui a servi à le réaliser et nous illustrerons avec l'expérimentation TRIBOT. Dans l'expérience TRIBOT de Mathias Paulin (Paulin, 2008)[5], le robot poursuit le biberon de façon réactive : tant qu'il est capable de produire un plan pour atteindre le biberon, il replanifie sans répit. On aimerait qu'il soit capable de prendre la décision de s'arrêter en fonction d'un apprentissage du comportement de son instructeur[6].

Dans notre expérience, on définit deux instructeurs : le bon instructeur et le mauvais instructeur.

Le but de cette expérience est d'utiliser les fonctionnalités d'apprentissage pour arriver à distinguer le bon instructeur du mauvais, et lorsque c'est le mauvais le TRIBOT doit décider l'action d'arrêter de poursuivre son biberon et de demander de l'aide à son instructeur.

Définition 1 (Comportement du bon instructeur) *Le bon instructeur ne déplace le biberon, trois fois au maximum, et* TRIBOT *arrive à attraper le biberon.*

Définition 2 (Comportement du mauvais instructeur) *Le mauvais instructeur déplace sans cesse le biberon (donc plus de trois fois), et le* TRIBOT *est systématiquement puni après.*

Les actions des instructeurs provoquent des perceptions enregistrées dans la mémoire épisodique du robot. Le but est que le robot parvienne à distinguer les deux comportements. Le robot peut ainsi apprendre à substituer au comportement compulsif de la recherche du biberon celui de la recherche d'une interaction avec son instructeur. Un tel mécanisme est considéré par les psychanalystes comme une étape nécessaire à la construction d'une pensée.

---

5. Video accessible sur http ://mathias.paulin.free.fr/
6. Video accessible sur http ://www.youtube.com/watch ?v=Rdajobt6laQ

2.2. *Système de l'action*

Les objets du système d'action sont commandées par les objets du système de diagnostic et par ceux du système de décision et elles agissent sur ceux du système de perception.

DECISION ⟶ ACTION
Action possible ⟶ Plan d'action

DIAGNOSTIC ⟶ ACTION
Action nécessaire ⟶ Plan d'action

Le diagnostic commande les actions réflexes et la décision les actions réfléchies. La décision et le diagnostic interviennent sur un système qui est en permanence en train d'agir. Le système devra décider ce qui doit se faire, continuer à se faire et ce qui peut se différer.

Le système est programmé en URBI qui gère avec une certaine autonomie l'agencement des tâches ; ce qui produit une certaine incertitude sur le fonctionnement du système. Comme une machine ayant une certaine autonomie n'est pas un automate, cette incertitude est un phénomène obligé.

Les actions du système consomment du temps et ne peuvent se réaliser qu'après certaines actions. Les actions se composent selon deux opérations : leurs exécutions en parallèle notée AVEC ou leurs exécutions successives notées PUIS. Soit le système est informé de devoir attendre que l'une et l'autre soient en état d'être exécutées ; on parle alors d'une loi de composition synchrone. Soit le système entreprend l'exécution d'une composition d'action quand l'une des actions est en état de se réaliser ; on parle d'un mode d'exécution asynchrone.

Deux modalités logiques président le choix des actions à faire celle de savoir si elle est "possiblement à faire" ou "nécessairement à faire". Le contrôle réflexe venant du diagnostic propose des séquences d'action à faire nécessairement. Alors que la décision va ordonner des actions possiblement à faire. De bonnes décisions vont engendrer des actions qui vont produire des résultats qui seront des perceptions attendues.

Les actions seront décidées par le robot ou par l'instructeur. Dans le cas de TRIBOT, nous avons des actions de l'instructeur et des actions du robot. Les actions de l'instructeur ne sont pas codées en URBI mais elle vont être traduites en perception. L'action de souffler dans le micro produit un reset du robot.

L'action de porter permet à l'instructeur d'agir sur l'apprentissage en le facilitant et en le perturbant.

Les actions du robot sous forme de programme URBI : Stopper puis alerter, Avancer lentement puis vite, Avancer lentement puis pincer, Chercher et Trouver.

Moteurs	Objet URBI
Moteur de la roue droite	`wheelR`
Moteur de la roue gauche	`wheelL`
Moteur de la pince	`claw`

Figure 4. *Equivalence entre les moteurs et leurs objets Urbi.*

2.3. *Système de perception*

Le système de perception du robot est composé des signifiants et des relations entre signifiants déclenchés par les séquences d'actions qu'elles viennent du robot ou de l'instructeur.

ACTION ⟶ PERCEPTION
séquence d'action ⟶ enregistrement de séquences d'actions
séquences d'action ⟶ valeur des capteurs

Pour le robot les signifiants sont les enregistrements à des instants donnés les valeurs des capteurs, les valeurs des capteurs après l'action et la désignation de l'action réalisée.

Capteur	Objet URBI
Capteur de distance	`sonar`
Capteur de niveau sonore	`sound`
Capteur de luminosité	`light`
Indicateur de batterie	`battery`

Figure 5. *Equivalence entre Capteurs et leurs objets Urbi.*

La mémoire épisodique est réactualisée en fonction des actions du robot. Elle se présente sous la forme d'un fichier `XML`. Il contient dans le préambule, le plan d'origine, la liste des actions utilisées et la liste des descripteurs. Dans la seconde partie, il contient l'exécution du plan, pour chaque action exécutée le cahier de labo contient les descripteurs pré et post action ainsi que la "prévision". Durant l'exécution du plan, si une re-planification a lieu, elle apparaît ainsi que le nouveau plan théorique.

Le résultat de l'expérience est + si le robot a réussi à saisir le biberon et - si il a été "puni" par l'instructeur.

2.4. *Système de diagnostic*

Nous allons développer davantage le système de diagnostic qui présente une originalité de cette étude.

PERCEPTION ⟶ DIAGNOSTIC
valeurs des capteurs ⟶ Variables

Les valeurs des capteurs du robot sont reformulées sous la forme de valeurs prises par des variables. Le diagnostic, va produire à partir de ces variables, des plans d'actions. Le diagnostic agit sur le système d'action du robot en lui proposant des comportements nécessaires : les relations entre les variables sont données par des contraintes.

DIAGNOSTIC ⟶ ACTION
instance de plan d'action nécessaire ⟶ Séquence d'actions

DIAGNOSTIC ⟶ DECISION
Instance de plan d'action possibles ⟶ décision

Le système va calculer en permanence des valeurs à attribuer aux actions pour permettre la réalisation de comportements possibles ou nécessaire.

Le choix de formuler par des contraintes un contrôle sur l'action, ne préjuge pas que le système soit assujetti à d'autres contraintes venant de calculs numériques comme cela se fait couramment en robotique. Dans le contexte de l'expérimentation TRIBOT, le diagnostic est uniquement réalisé par l'usage de contraintes. Le logiciel de programmation par contrainte utilisé est JCHOCO.

2.4.1. *Programmation par contraintes*

La programmation par contraintes a été formalisée en 1974 par Montanari (Montanari, 1974) et en 1977 par Mackworth (Mackworth, 1977). La programmation par contraintes est utilisée pour résoudre des problèmes d'affectation de valeurs à des variables. Ces variables sont liées entre elles par des relations que l'on nomme contraintes. Cela revient à poser un problème sous forme de relations logiques entre plusieurs variables. Un problème à résoudre comporte un ensemble fini de variables. Chacune des variables possède un domaine et un certain nombre de contraintes, le domaine des variables peut être discret ou continu.

Dans notre expérimentation, on ne manipule que des réseaux de contraintes discrets.

Définition 3 (Réseau de contraintes) *Un réseau de contraintes $\mathcal{R}$ est un triplet $(\mathcal{X}, \mathcal{D}, \mathcal{C})$ où :*

– $\mathcal{X} = \{x_1, \ldots, x_n\}$ *est l'ensemble des variables du problème ;*

– $\mathcal{D} = \{d(x_1), \ldots, d(x_n))$ est l'ensemble des domaines des variables du problème. Chaque variable x prend ses valeurs dans $d(x)$;

– $\mathcal{C} = \{c_1, \ldots, c_m\}$ est un ensemble de contraintes sur $\mathcal{X}$. Une contrainte c est définie un ensemble de variables que l'on note $var(c)$, et une relation sur ses variables que l'on note $rel(c)$ qui spécifie les tuples autorisés par la contrainte.

Une contrainte est dite binaire lorsqu'elle implique exactement deux variables. Un réseau de contraintes est dit binaire lorsque l'ensemble $\mathcal{C}$ ne contient que des contraintes binaires.

Définition 4 (Instanciation) *Soit $\mathcal{Y} = \{y_1, \ldots, y_k\}$ où $\mathcal{Y} \subseteq \mathcal{X}$ une instanciation $E_\mathcal{Y}$ de $\mathcal{Y}$ est un k-uplet $(v_1, \ldots, v_k) \in (D_1 \times \ldots \times D_k)$. On dit que l'instanciation est complète lorsque $\mathcal{Y} = \mathcal{X}$ (cette instanciation sera notée E).*

Une instanciation $E_\mathcal{Y}$ viole la contrainte $c \in \mathcal{C}$ si et seulement si

$$var(c) \subseteq \mathcal{Y}, E_\mathcal{Y}\left[var(c)\right] \notin rel(c)$$

Définition 5 (Solution) *Une solution est une instanciation complète telle que $\forall c \in \mathcal{C}, E\left[var(c)\right] \in rel(c)$. On note l'ensemble des solutions d'un réseau $S(\mathcal{R})$, ou $S(\mathcal{X}, \mathcal{D}, \mathcal{C})$.*

2.4.2. *Planification d'action*

La planification d'action proposée est une extension du modèle STRIPS.

STRIPS (ou STanford Research Institute Problem Solver) est un algorithme de Planification classique conçu par Richard Fikes et Nils Nilsson en 1971 (Fikes *et al.*, 1971). On nomme aussi par ce nom le langage de représentation des données utilisée par l'algorithme. Avec le GPS (General Problem Solver) de Newell et Simon de 1961, il fait partie des premiers planificateurs utilisés en intelligence artificielle et été suivi de nombreux dérivés (GraphPlan, IPP, STAN, etc).

Dans le module de supervision de Mathias Paulin (Paulin, 2008), les états initiaux et finaux sont contraints par des contraintes d'égalité. Cette représentation des états limite les buts que l'on peut donner au robot, pour palier à ce problème nous définissons les méta-contraintes.

Afin de pouvoir poser des contraintes plus variées sur les états initiaux et finaux (c'est à dire plus variées que la simple égalité), nous avons défini un ensemble de méta-contraintes qui sont instanciées par le solver. Elles portent sur les variables de l'état final ou initial du réseau de contraintes représentant le problème de planification.

Définition 6 (pre(a_i)) *$pre(a_i)$ représente l'ensemble des préconditions de l'action a_i.*

$$pre(a_i) = \{c_0, \ldots, c_k\}$$

où k est le nombre de contraintes composant la précondition.

Définition 7 (post(a_i)) *$post(a_i)$ représente l'ensemble des postconditions de l'action a_i.*

$$post(a_i) = \{c_0, \ldots, c_k\}$$

où k est le nombre de contraintes composant la postcondition.

On préférera représenter les contraintes comme un ensemble de contraintes binaires plutôt que comme une conjonction afin de pouvoir plus facilement filtrer des contraintes notamment lors de la compilation de séquences d'action.

Définition 8 (modified(a_i)) *$modified(a_i)$ représente l'ensemble des descripteurs modifiés par l'action a_i.*

Définition 9 (Méta-Contrainte) *On peut voir une méta-contrainte comme une fonction qui associe un état $\mathcal{E}$, un ensemble de descripteurs $D = d_0, \ldots, d_n$, ainsi qu'une contrainte et qui associe une contrainte portant sur les variables de l'état $\mathcal{E}$ correspondant à l'ensemble de descripteur D.*

$$metacontrainte(\mathcal{E}, \mathcal{D}_0, \ldots, \mathcal{D}_n) : \mathcal{E} \times \mathcal{D}^n \rightarrow \mathcal{C}$$
$$(e, (d_0, \ldots, d_n)) \rightarrow c(\mathcal{E}(d_0, \ldots, d_n))$$

2.4.2.1. Définition des Variables

Pour modéliser le problème de planification de longueur k nous définissons $k+1$ ensembles de variables descripteurs que nous nommerons états.

L'état à l'étape s, $\mathcal{E}_s$ est définit ainsi $\mathcal{E}_s = \{d_0^s, \ldots, d_n^s\}$ ou d_0^s représente le descripteur 0 à l'étape s, le domaine de chaque variable descripteur $d_i \in [min(d_i), max(d_i)]$. L'ensemble des actions $\mathcal{A}$ est $\{a_0, \ldots, a_m\}$. Pour chaque étape du plan s, on définit une variable $action_s$ dont le domaine est l'ensemble des indices des actions contenues dans $\mathcal{A}$ ($a_s \in [0, m]$). L'assertion $action_s = index(a_e)$ est vraie si l'action a_e permet la transition entre $\mathcal{E}_s$ et $\mathcal{E}_{s+1}$, donc entre les étapes s et $s+1$ du plan.

2.4.2.2. Définition des Contraintes

Tout d'abord il faut imposer des contraintes sur l'état initial $\mathcal{E}_i$ et l'état final $\mathcal{E}_f$ ($i = 0$ et $f = k+1$). Ici l'état initial nous est donné par l'utilisateur (Etat du robot, ou état du monde) sous forme d'un ensemble de méta-contraintes, ainsi on instancie les méta contraintes portant sur l'état initial[7]. Il faut également instancier les méta contraintes portant sur l'état final.

7. On notera que si un état initial n'est pas suffisamment contraint, le solveur choisira des valeur pour les variables de l'état initial et donc mènera à des résultats inattendus

Ensuite pour chaque étape, une action $a_i \in \mathcal{A}$ ne peut être exécutée entre les pas s et $s+1$ que si sa précondition sur $\mathcal{E}_s$ est vraie. Comme on veut être en mesure d'exécuter une action à toute étape du plan, pour chaque action nous ajoutons la contrainte $(a_s = i) \Rightarrow pre(a_i, \mathcal{E}_s)$ (la précondition de l'action a_i appliquée à l'état $\mathcal{E}_s$).

Pour finir, on instancie les exclusions mutuelles $Mutex$(Action a_1, Action a_2, Integer $ordre$) sous la forme $(a_i = a_1) \implies (a_{i+ordre} \neq a_2)$

Ainsi défini le modèle est semblable au Base-CSP défini par Lopez (Lopez *et al.*, 2003) avec une extension aux variables non booléennes. Les architectures de planification et de supervision sont capables de planifier et d'exécuter des actions en parallèle.

La résolution de ce réseau de contraintes nous permet d'obtenir une séquence d'action permettant au robot d'atteindre le but fixé.

2.5. *Le contrôle réactif*

Le contrôle réactif s'effectue selon la boucle impliquant les systèmes de diagnostic, d'action et de perception qui caractérise une architecture à la Brooks..

DIAGNOSTIC $\longrightarrow$ ACTION $\longrightarrow$ PERCEPTION $\longrightarrow$ DIAGNOSTIC

Le contrôle réactif est réalisé en traduisant des plans nécessaires directement en séquences d' action sans passer par la phase de décision.

Le plan prévu est réactualisé lors de cette boucle. Quand le plan est perturbé par l'instructeur ou par une erreur des capteurs, le système effectue une nouvelle passe de planification.
Deux mécanismes sont en place afin de permettre au robot d'être plus efficace :

– La parallélisation des actions. En effet la durée d'exécution du plan s'en retrouve plus courte et le robot dépense moins d'énergie ;

– La compilation d'actions fréquentes permet de diminuer la complexité de la passe de la planification ;

2.6. *Système de décision*

Le système de décision est influencé par les systèmes de perception et de diagnostic. Il doit gérer toute la supervision du robot par lui-même comme par son instructeur.

DIAGNOSTIC ⟶ DECISION
variables ⟶ termes
plan d'action possible ⟶ plan d'action

PERCEPTION ⟶ DECISION
Ensemble d'expériences ⟶ règles de décision

Ce système est un système expert permettant de formuler de règles sur des comportements exprimés par des séquences d'actions. Les règles du systèmes peuvent être fournie par le constructeur du système, par les instructeurs ou encore par le robot lui-même qui les produit par apprentissage.

La logique du système intègre est inspirée par la logique juridique dont les lois, et les normes régulent des comportements tout en permettant la présence d'aléa et de défaillances de la part des acteurs. Les objets d'integre constituent une ontologie de termes hiérarchisés et définis par des graphes conceptuels. Les relations d'integre sont des ensemble de règles logiques. Une instance en intègre est une suite de termes instanciés. Un plan est une instance dans integre. La supervision d'un plan va consister à en déterminer des contradictions et des incomplétudes.

Donnons un aperçu du réseau sémantique impliqué dans la prise de décision du Tribot (voir *figure* 6). Quand le plan est d'attraper et que le biberon est entre 20 et 60 cm, si l'état précédent du comportement est d'approcher et que la cible est à plus 30 cm alorsle comportement est d'approcher. Si la cible est à moins de 30cm l'état du comportement est d'attraper. Mais si la cible s'éloigne, l'état du comportement redevient "approcher".

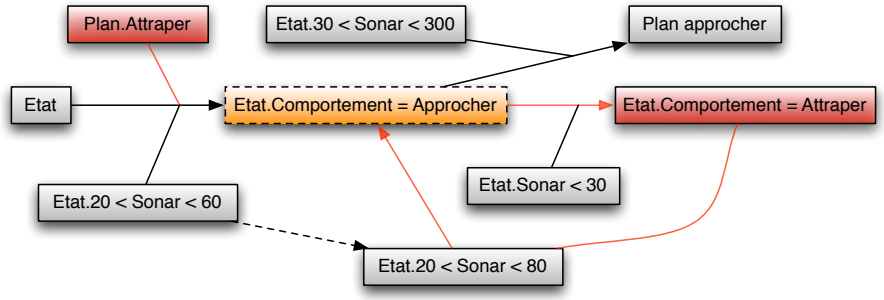

Figure 6. *Un fragment du réseau sémantique de la prise de décision*

2.7. *L'apprentissage*

L'apprentissage intervient dans la réalisation de la boucle comportementale. Le

ACTION ⟶ PERCEPTION ⟶ DECISION ⟶ ACTION

robot dispose de sa mémoire épisodique pour apprendre des règles de comportement. De nombreuses méthodes d'apprentissage sont candidates à réaliser cette boucle.

Par souci de simplicité, nous avons utilisé WEKA(Witten *et al.*, 2005) afin de pouvoir tester différents classificateurs.

Comme la règle à apprendre est de type fréquentielle, (on veut que le robot stoppe sa recherche de biberon lorsqu'il l'a déjà cherché trois fois (ie. $frequence(FindTarget) > 3$)). On transforme donc chaque expérience de la mémoire épisodique en un vecteur de fréquence dont la taille est le nombre d'action du robot, et ou chaque élément est le nombre de fois où l'action est apparue durant l'expérience.

On transforme la mémoire épisodique en un fichier ARFF dont les instances sont composées de la fréquence d'apparition de chaque action à l'intérieur d'une même expérience et du résultat de cette expérience. Le résultat de l'expérience représente la classe de l'instance.

Dans cette section, nous avons décrit comment les sous-systèmes interagissent dans le cadre de cette architecture. Sur l'exemple TRIBOT, nous avons montré le genre d'apprentissage que nous pourrions réaliser ; apprentissage inspiré des mécanismes de psychisation et décrits en pédopsychiatrie.

3. Conclusion

Nous proposons une architecture générique pour caractériser un système autonome prenant en compte l'interaction, le contrôle réactif et l'apprentissage pour améliorer ses performances.

Cette architecture est indépendante de la réalisation matérielle et elle permet à un système réalisé matériellement d'accroître sa capacité de donner de la signification à ses actions.

Ce système est construit autour de trois boucles combinant quatre sous systèmes. Les deux mécanismes, l'apprentissage et le contrôle réactif, se conjuguent pour réaliser l'adéquation du système.

Cette architecture est bio-inspirée comme l'ont montré les nombreux exemples. Les outils informatiques proposés pour la réalisation du système informatique sont génériques et ils sont sous exploités dans l'application réalisée qui n'avait comme unique enjeu que de montrer que l'on pouvait réaliser effectivement cette architecture.

Ce travail ouvre des perspectives sur deux plans : le plan formel et le plan de sa signification en psychologie.

Sur le plan formel, le système présenté est susceptible d'une définition formelle en théorie des catégories. Une formalisation catégorielle sert à établir des correspondances avec d'autres formalisations ; ainsi a été montré la correspondance entre les différents formalismes du calcul en informatique théorique. Un tel travail reprendrait le projet de Lacan de définir par des mathèmes des structures opératoires du psychisme.

Dans cet état d'esprit, il est possible de rechercher quelle logique intuitionniste correspond à ce modèle. Chaque système à ses propres modalités : il y aurait des modalités permettant de gérer le temps (système de l'action), le risque (système de la décision), la génération d'hypothèses (système du diagnostic) et les modalités existentielles (système de perception).

Sur le plan de sa signification psychologique, nous comptons monter des expérimentations apportant une meilleure connaissance du processus de psychisation tant pour les humains que pour les machines.

4. Bibliographie

Barthes R., *Le Neutre, Cours au Collège de France*, 1977-1978.

Bauer A., Klasing K., Lidoris G., Mühlbauer Q., Rohrmüller F., Sosnowski S., Xu T., Kühnlenz K., Wollherr D., Buss M., « The Autonomous City Explorer : Towards Natural Human-Robot Interaction in Urban Environments », 2009.

Bourreau E., Peralta E., Sallantin J., « Scientific Discovery of itself by a robot assisted by an human », *ECAP 09*, J. Vallverdu, 2009.

Brooks R. A., A Robust Layered Control System For a Mobile Robot, Technical report, Cambridge, MA, USA, 1985.

Coeeckelgergh M., « Robot anthropology : Robots as hermeneutic devices for defining human », *ECAP 09*, J. Vallverdu, 2009.

Cohen P. R., Levesque H. J., « Intention is Choice with Commitment », *Artificial Intelligence*, vol. 42, n°: 2-3, p. 213-261, 1990.

Dor J., *Introduction à la lecture de Lacan : l'inconscient structuré comme un langage*, Espace Analytique, Denoel, 2002.

Fikes R., Nilsson N. J., « STRIPS : A New Approach to the Application of Theorem Proving to Problem Solving », *IJCAI*, p. 608-620, 1971.

Golse B., *Du corps à la pensée*, Le fil rouge, PUF, 1999.

Golse B., *L'être-bébé*, Le fil rouge, PUF, 2007.

Ishii K., « Emerging conciousness and shizophrenic byproducts in Humanoid Robots », *ECAP 09*, J. Vallverdu, 2009.

Lipoubou L., Le fantôme dans la machine, PhD thesis, Université Montpellier II, 2005.

Lopez A., Bacchus F., « Generalizing GraphPlan by Formulating Planning as a CSP », *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-2003), 954960*, Morgan Kaufmann Publishers, p. 954-960, 2003.

Mackworth A., « Consistency in Networks of Relations », *Artificial Intelligence*, vol. 8, n°: 1, p. 99-118, 1977. Reprinted in Readings in Artificial Intelligence, B. L. Webber and N. J. Nilsson (eds.), Tioga Publ. Col., Palo Alto, CA, pp. 69-78, 1981. [This paper was honoured in Artificial Intelligence 59, 1-2, 1993 as one of the fifty most cited papers in the history of Artificial Intelligence.].

Minsky M., *La société de l'esprit*, Interédition, 1988.

Montanari U., « Networks of constraints : Fundamental properties and application to picture processing. », 1974.

Oudeyer P.-Y., Kaplan F., Hafner V., « Intrinsic Motivation Systems for Autonomous Mental Development », *IEEE Transactions on Evolutionary Computation*, 2007.

Parrochia D., *Qu'est-ce que penser/calculer ? : Hobbes, Leibniz et Boole*, J. Vrin, 1992.

Paulin M., Contributions à l'apprentissage automatique de réseau de contraintes et à la constitution automatique de comportements sensorimoteurs en robotique, PhD thesis, Université Montpellier II, 2008.

Witten I. H., Frank E., *Data Mining : Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*, Morgan Kaufmann, June, 2005.