Institut National Polytechnique de Grenoble

N° attribué par la bibliothèque

|__/__/__/__/__/__/__/__/__/__/

THÈSE

pour obtenir le grade de

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Discipline : Sciences Cognitives

Présentée et soutenue publiquement par

OLIVIER LEBELTEL

Le 8/10/1999

PROGRAMMATION BAYÉSIENNE DES ROBOTS

Co-directeurs de thèse
Pierre Bessière et Emmanuel Mazer

COMPOSITION DU JURY

M. Pierre Escudier	Président
M. Rachid Alami	Rapporteur
M. Paul Bourgine	Rapporteur
M. Emmanuel Mazer	Directeur de thèse
M. Pierre Bessière	Co-directeur de thèse
M. Dominique Luzeaux	Examinateur

Thèse préparée au sein du laboratoire LEIBNIZ GRENOBLE

Remerciements

Je tiens tout d'abord à exprimer ma reconnaissance envers mes deux directeurs de thèse et amis, Emmanuel Mazer et Pierre Bessière, dont l'influence, le soutien et l'amitié m'ont permis de mener à bien ce travail.

Je souhaite adresser ma plus vive gratitude aux membres de mon jury. Je remercie Monsieur Pierre Escudier, directeur de recherche au CNRS, directeur de l'Institut de la Communication Parlée (Grenoble), pour m'avoir fait l'honneur de présider le jury. Je remercie mes rapporteurs pour le travail que leur a demandé ma thèse : Monsieur Rachid Alami, Directeur de Recherche au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS (LAAS Toulouse) et Monsieur Paul Bourgine, Ingénieur IGREF, enseignant chercheur au Centre de Recherche en Épistémologie Appliquée de l'École Polytechnique. Je remercie également Monsieur l'Ingénieur Principal de l'Armement Dominique Luzeaux du Centre Technique d'Arcueil (CTA/GIP) pour l'intérêt qu'il a porté à ce travail.

J'adresse également mes remerciements à Monsieur Philippe Jorrand, directeur du laboratoire Leibniz pour m'avoir accueilli au sein de son laboratoire. Je tiens aussi à remercier tous les membres de l'équipe Laplace, Pascal di Giacomo, Juan-Manuel Ahuactzin-Larios, Eric Dedieu, Remis Balaniuk, Kamel Mekhnacha, Réda Berrah, Julien Diard et Cyrille Lefranc. J'ai connu au sein de cette équipe une formidable ambiance, des discussions passionnées et l'impression de vivre une réelle aventure scientifique.

Toute au long de ce travail, j'ai eu la grande chance de bénéficier de l'enseignement et de la fraternité de nombreuses personnes. Il serait trop long de les citer toutes ici. Malgré l'incertitude résultant de ces remerciements incomplets, ma gratitude à leur égard est grande, même si elle n'est pas explicitement exprimée.

Enfin, je remercie ma femme Agata et nos enfants Jules, Lola et Anita, pour leur patience durant ces longues années pendant lesquelles leur papa écrivait « son gros livre plein de lettres » et répondait régulièrement à la question « Alors, t'as fini ta thèse cette fois ci ? » par un « Bientôt, … encore une petite quinzaine … ».

À mon père.

Sommaire

Remerciements 3
Sommaire 7
Table des figures 11
Introduction 13
1.1 Situation 13
1.2 Contribution 14
1.3 Plan de lecture 14
Chapitre 2 Concepts de base 17
2.1 Définitions et notations 17
2.2 Règles de calcul 20
2.3 Méthode 24
2.4 Programme et interpréteur probabiliste 31
Chapitre 3 Présentation de la plate-forme expérimentale 35
3.1 Le Khepera 36
3.2 L'environnement 38
3.3 Contrôle des flux sensoriels et moteurs 39
3.4 Exécution d'un programme probabiliste. 40
3.5 Télé-opération 41
Chapitre 4 Comportements réactifs 43
4.1 Phototaxie 43

4.2 Apprentissage de comportements réactifs 48
4.3 Spécification a priori de comportements réactifs 58
Chapitre 5 Modèle capteurs *75*
5.1 Objectifs et protocole expérimental 75
5.2 Phase 1 : modèle capteur 76
5.3 Phase 2 : fusion des capteurs 84
5.4 Enseignement et discussion 93
Chapitre 6 Combinaison de comportements *95*
6.1 Tourner et ralentir en présence d'obstacles 95
6.2 Se diriger vers une source lumineuse en évitant les obstacles 104
6.3 Enseignements et discussion 113
Chapitre 7 Composition hiérarchique de descriptions *115*
7.1 Objectifs et protocole expérimental 115
7.2 Spécification 115
7.3 Identification 117
7.4 Utilisation 118
7.5 Enseignements et discussion 123
Chapitre 8 Reconnaissance de situations *125*
8.1 Reconnaissance de la base 125
8.2 Reconnaissance d'objets 130
8.3 Enseignements et discussion 136
Chapitre 9 Séquences temporelles *137*
9.1 Objectifs 137
9.2 Séquence temporelle de tâche 137
9.3 Séquence temporelle de déplacement 145
Chapitre 10 Intégration : une tâche de surveillance *151*
10.1 Objectifs 151
10.2 Les descriptions disponibles 151
10.3 Spécification 153
10.4 Identification 155
10.5 Utilisation 157
10.6 Enseignements et discussion 165
Chapitre 11 Synthèse *167*
11.1 Situation 168
11.2 Contributions 175
11.3 Perspectives 181

11.4 Conclusion ____ **187**

Bibliographie ____ *189*

Annexe A ____ *197*

A.1 Introduction ____ **200**

A.2 Critique de l'approche F+I des systèmes sensori-moteurs ____ **204**

A.3 Introduction à l'approche F+D des systèmes sensori- moteurs ____ **215**

A.4 Une perspective en guise de conclusion : l'évolution contingente des connaissances préalables ____ **230**

A.5 Bibliographie ____ **236**

Annexe B ____ *239*

B.1 PaL : un fondement mathématique pour l'approche F+D ____ **242**

B.2 Raisonner malgré l'incertitude et l'incomplétude : l'inférence probabiliste. ____ **243**

B.3 Ancrer les descriptions : le principe de maximum d'entropie ____ **248**

B.4 Notes bibliographiques ____ **258**

B.5 Bibliographie ____ **259**

Table des figures

Figure 3-1 — Schéma de la plate-forme expérimentale. 35
Figure 3-2 — Le Khepera en action. 36
Figure 3-3 — Schéma du Khepera 36
Figure 3-4 — L'environnement du Khepera 38
Figure 3-5 — Schéma du squelette type d'un programme de contrôle du robot. 39
Figure 4-1 — Variables pertinentes pour le comportement phototaxie. 44
Figure 4-2 — Visualisation de P(`Vrot` | `Lum` Cp_Photo) pour les différentes valeurs de `Lum`. 46
Figure 4-3 — Pré-traitements sur les capteurs de proximité avant, ici `Dir` = -4 et `Prox` = 3. 49
Figure 4-4 — L'effet d'un modèle gaussien sur un phénomène bi-modal. 54
Figure 4-5 — Apprentissage par télé-opération d'un comportement de poussé d'obstacle 55
Figure 4-6 — Restitution de l'apprentissage d'un comportement de poussé d'obstacle. 56
Figure 4-7 — Apprentissage par télé-opération d'un comportement de suivi de contour. 56
Figure 4-8 — La fonction sigmoïde σ_r(`Prox`). 59
Figure 4-9 — La fonction μ_r(`Dir`, `Prox`). 60
Figure 4-10 — P(`Vrot` | `Dir` `Prox`) pour `Dir`=7 et différentes valeurs de `Prox`. 61
Figure 4-11 — P(`Vrot` | `Dir` `Prox`) pour `Prox`=15 et différentes valeurs de `Dir`. 61
Figure 4-12 — Le Khepera placé devant un couloir. 63
Figure 4-13 — Évolution des variables `Dir` et `Prox` lors du parcours d'un couloir. 63
Figure 4-14 — Évolution de μ_t(`Prox`,`Vtrans_c`) pour `Vtrans_c` = 7. 66
Figure 4-15 — Robot débouchant d'un couloir. 68
Figure 4-16 — Approximation probabiliste d'une loi de contrôle. 72
Figure 5-1 — Choix des variables pertinentes pour le modèle de la réponse d'un capteur de luminosité. 76
Figure 5-2 — Profil de la réponse d'un capteur de luminosité 78
*Figure 5-3 — Fonction **LM**(`DistL`).* 78
*Figure 5-4 — Fonction **K**(`ThetaL`).* 79
Figure 5-5 — Connaissance sur l'orientation et la distance d'une source lumineuse. 82
Figure 5-6 — Connaissance sur l'orientation et la distance d'une source lumineuse. 83
Figure 5-7 — Connaissance sur l'orientation de la source lumineuse. 89
Figure 5-8 — Connaissance sur l'orientation de la source lumineuse. 89
Figure 5-9 — Réponse prédite du capteur `Lm3`. 91
Figure 5-10 — Fusion des huit capteurs pour `Theta` = 10 et `DistL` = 20. 92
Figure 5-11 — Fusion des huit capteurs pour `Theta` = 10 et `DistL` = 10. 92
Figure 6-1 — La dépendance P([`H=e`] | `Prox` C) spécifiée comme fonction de `Prox` par une sigmoïde. 107

Figure 6-2 — Combinaison évitement d'obstacle et phototaxie. _____ *110*
Figure 6-3 —Combinaison des comportements « phototaxie » et « évitement d'obstacle ». _____ *111*
Figure 6-4 — Combinaison phototaxie et évitement d'obstacle : `Lum` *= -90,* `Dir` *= -5,* `Prox` *= 10.* _____ *112*
Figure 6-5 — Combinaison phototaxie et évitement d'obstacle : `Lum` *= -90,* `Dir` *= -5,* `Prox` *= 10.* _____ *112*
Figure 6-6 — Combinaison phototaxie et évitement d'obstacle : `Lum` *= -90,* `Dir` *= 5,* `Prox` *= 8.* _____ *113*
Figure 7-1 — La fonction σ_p*(*`ThetaL`*).* _____ *117*
Figure 7-2 — La fonction sigmoïde μ_p*(*`ThetaL`*).* _____ *118*
Figure 7-3 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle. _____ *122*
Figure 7-4 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle. _____ *122*
Figure 7-5 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle. _____ *123*
Figure 7-6 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle. _____ *123*
Figure 8-1 — Relation de dépendances entre les 16 variables sensorielles et la variables `Base`*.* _____ *126*
Figure 8-2 — Retour à la base. _____ *130*
Figure 8-3 — Les différents objets présentés au Khepera. _____ *135*
Figure 10-1 — Structure de la question permettant d'inférer `Vrot` *et* `Vtrans`*.* _____ *161*
Figure 10-2 — Structure de la question permettant de calculer Vrot et Vtrans. _____ *163*
Figure 10-3 — Tâche de surveillance. _____ *164*
Figure 11-1 — L'approche symbolique en robotique. _____ *168*
Figure 11-2 — L'approche constructiviste en robotique. _____ *170*
Figure 11-3 — L'approche suivie dans cette thèse _____ *171*
Figure 11-4 — Fondements théoriques : « de l'incomplétude à la décision » _____ *176*

Introduction

1.1 Situation

Chacun ayant eu à programmer un robot réel dans un environnement physique s'est inévitablement retrouvé confronté aux problèmes de l'incertitude. Les valeurs obtenues sur les capteurs sont « bruitées », les conséquences des commandes motrices ne sont jamais précisément celles escomptées, les modèles sont entachés « d'erreurs »… En robotique, la nécessité de gérer la dimension incertaine est incontournable.

La position soutenue dans cette thèse est qu'il est nécessaire et avantageux d'adopter une démarche formelle, rigoureuse et systématique pour exprimer, traiter et débattre de cette dimension incertaine.

La spécificité de notre approche repose sur notre conviction que l'origine de cette incertitude relève avant tout de l'inévitable incomplétude des modèles utilisés pour décrire une tâche robotique. Il existe toujours des facteurs non pris en compte (variables cachées) qui font que le phénomène et le modèle ne se comportent jamais exactement de la même manière.

Cette position est le fruit d'une réflexion plus générale sur la perception, l'inférence et l'action menée au sein de l'équipe LAPLACE du laboratoire LEIBNIZ. Le cadre théorique adopté dans cette équipe est celui de l'inférence et l'apprentissage bayésien. L'apprentissage et les techniques de maximum d'entropie permettent de transformer l'incomplétude (ce qui n'a pas été explicitement modélisé) en incertitude. L'inférence bayésienne prend ensuite le relais et permet de raisonner avec cette incertitude. Le champ d'expérimentation et d'application privilégié retenu par l'équipe est la robotique. La robotique pose de manière incontournable les problèmes réels des systèmes sensori-moteurs. Cette réflexion est décrite en détail dans la thèse d'Eric Dedieu (voir [Dedieu95]) et dans les deux articles ([Bessière98a] et [Bessière98b]) insérés en annexe.

1.2 Contribution

Le résultat de cette thèse est de proposer un système original de programmation pour la robotique autonome, fondé sur cette approche bayésienne. L'apport principal de ce travail est une méthode générique reposant sur la notion centrale de *description*.

Dans le cadre de cette méthode, une tâche robotique est représentée sous la forme d'une relation de dépendance probabiliste (*distribution conjointe*) entre différentes variables choisies par le programmeur. Cette relation n'est pas quelconque ; elle est le fruit d'un ensemble de connaissances préalables fournies par le programmeur et d'un ensemble de données expérimentales. La *description* est l'objet formel faisant la synthèse de ces connaissances préalables et de ces données expérimentales. Elle peut être interrogée pour réaliser une tâche robotique, par exemple, afin de déterminer la valeur de certaines variables motrices, sachant la valeur de certaines variables sensorielles. La *description* peut aussi être utilisée dans des raisonnements complexes régis par les règles de l'inférence bayésienne.

Pour programmer et réaliser une tâche robotique, la démarche proposée est systématique et se décompose en trois phases :

- *Phase de spécification* où le programmeur donne les connaissances préalables par :
  - Le choix des variables pertinentes ;
  - Le choix de la décomposition de la distribution conjointe en distributions élémentaires ;
  - Le choix de la forme paramétrique associée à chacune de ces distributions élémentaires.
- *Phase d'identification* où les valeurs des paramètres libres des différentes distributions élémentaires sont déterminées. Ces valeurs sont soit le résultat d'une identification à partir de données expérimentales, soit fixées a priori, soit le résultat de l'interrogation d'une autre description.
- *Phase d'utilisation* où la description est interrogée en posant des questions probabilistes. Par exemple, dans le cas du contrôle moteur, nous calculons la probabilité des valeurs de différentes variables motrices connaissant les valeurs des variables sensorielles. Nous obtenons ainsi une distribution de probabilité sur les valeurs de ces variables motrices. Nous déterminons les ordres à transmettre au robot en tirant au hasard selon cette distribution de probabilité.

Cette méthode présente de nombreux points forts. Elle offre l'avantage d'expliciter formellement les connaissances préalables du programmeur, les processus adaptatifs mis en jeu et de restituer l'incomplétude de ces informations sous forme d'incertitude. Elle présente de plus une homogénéité rigoureuse pour la spécification et l'utilisation des programmes et à ce titre acquiert un caractère général et systématique.

1.3 Plan de lecture

Notre objectif est de présenter notre méthode comme un cadre formel permettant de programmer des robots. Nous avons choisi de suivre un cheminement progressif, jalonné d'expériences de plus en plus riches, illustrant sa faisabilité et son intérêt pratique. Au long de sept chapitres nous décrirons plusieurs programmes visant à contrôler un robot mobile pour des tâches simples au départ, puis de plus en plus complexes. Au fur et à mesure des expériences, suivant une démarche constructive, nous enrichirons les ressources disponibles

pour programmer notre robot en abordant différentes facettes de la programmation : comportements réactifs, modèles capteur, combinaisons de comportements, reconnaissance de situation, enchaînement de tâches, apprentissage... Dans ces chapitres, notre ambition est de mettre en évidence le caractère formel et systématique de notre approche. Pour cela, chaque expérience sera toujours exposée suivant le même schéma composé des sections suivantes :

- *Objectif et protocole expérimental* : Ici, pour chaque expérience, nous décrirons les objectifs de la tâche robotique et les conditions particulières dans lesquelles elle sera réalisée.
- *Spécification* : Cette section présentera le choix des connaissances préalables de la description. Nous détaillerons les variables retenues, le choix de la décomposition de la distribution conjointe et les formes paramétriques associées.
- *Identification* : Dans cette section nous présenterons comment sont fixées les valeurs des distributions composant la description.
- *Utilisation* : Là, nous décrirons le programme de contrôle et les résultats obtenus. Nous indiquerons la (ou les) description mise en œuvre et la requête probabiliste associée.
- *Enseignement et discussion* : Enfin nous conclurons chaque expérience en discutant des points importants à retenir.

Nous attendrons le dernier chapitre pour prendre du recul, revenir sur les principes fondamentaux, les discuter et donner quelques éléments bibliographiques. Nous espérons qu'alors, en s'appuyant sur les expériences concrètes décrites auparavant, les notions nouvelles et les principes fondamentaux pourront être exposés et débattus beaucoup plus clairement et simplement.

Ce document est donc structuré de la manière suivante :

- Les chapitre 2 et 3 présentent les bases mathématiques et la plate-forme expérimentale. Ils sont indispensables pour comprendre les expériences qui suivent.
- Nous abordons ensuite une série d'expériences destinées à montrer comment on peut définir, programmer et apprendre au robot des comportements réactifs (Chapitre 4). Nous présentons différents comportements réactifs et la manière dont nous les avons réalisés sous la forme de *descriptions*. L'objectif poursuivi est de fournir au robot Khepera un ensemble de comportements simples lui permettant d'interagir avec son environnement. L'identification des formes paramétriques sera obtenue à partir de données expérimentales
- Le chapitre suivant (Chapitre 5) décrit comment on peut construire un modèle bayésien d'un capteur et, surtout, comment on peut fusionner les informations venant d'un ensemble de capteurs
- Le chapitre 6 décrit deux expériences montrant comment combiner entre eux des comportements déjà appris pour élaborer des comportements plus complexes.
- Le chapitre 7 montre comment on peut hiérarchiquement construire des descriptions probabilistes de comportements de plus en plus abstraits. Nous illustrons ce point par la composition de la description d'un modèle capteur avec la description d'une loi de contrôle réactive.
- Le chapitre suivant illustre comment aborder les problèmes de classification et de reconnaissance (chapitre 8). Le robot pourra détecter lorsqu'il se trouve à sa base et reconnaître différents objets en suivant leur contour.

- Le chapitre 9, illustre la construction de séquences temporelles et introduit une notion originale de programmation inverse.
- Le chapitre 10 expose en détail une tâche de robotique mobile complexe intégrant les différentes descriptions construites dans les chapitres précédents. Cette dernière expérience présente l'utilisation de l'ensemble de ces ressources dans le cadre d'une tâche de surveillance. Ici, notre objectif sera tout d'abord de montrer comment l'expression d'une tâche complexe robotique peut être obtenue sous la forme d'une *description*. Nous montrons également que cette description peut-être construite sous la forme d'une « structuration » de descriptions plus élémentaires, et en particulier, nous illustrons comment nous pouvons utiliser les différentes descriptions, spécifiées tout au long des chapitres précédents, comme briques de base pour programmer.
- En guise de conclusion, une synthèse sera proposée afin de discuter plus globalement de notre approche et de sa situation au sein de la robotique et des sciences cognitives. Nous reviendrons sur les éléments mathématiques de la théorie sous-jacente et sur les principaux points techniques et méthodologiques. Cette discussion tentera d'éclairer le problème de la programmation d'un système artificiel en situation réelle. Nous situerons notre travail avec quelques références bibliographiques en opposant les deux grandes approches en robotique : l'approche « classique » relevant de la « cognition de haut niveau » et l'approche « réactive » associée à une « cognition de bas niveau ». Nous montrerons finalement comment nos travaux proposent de faire le lien entre ces deux extrêmes.

Par souci de clarté sur notre contribution proprement personnelle, il a été choisi de joindre en annexe deux articles collectifs, qui reviennent de manière approfondie sur les fondements de notre approche et élargissent les perspectives sur la situation de ce travail à l'intérieur des sciences cognitives et de la robotique, notamment en complétant les références bibliographiques données dans la thèse.

Chapitre 2

Concepts de base

Ce chapitre a pour but de présenter les bases théoriques et mathématiques minimums nécessaires à la compréhension de ce travail. De nombreuses questions laissées en suspend trouveront donc progressivement leurs réponses lors de la présentation des expériences et au cours de la discussion finale.

2.1 Définitions et notations

2.1.1 Propositions logiques

Les objets manipulés sont des propositions de la logique classique. Elles peuvent être soit vraies soit fausses. La valeur de vérité de ces propositions doit être définie de manière non ambiguë. Ainsi nous considérons que l'expression « La taille de Jean est comprise entre 1 m et 2 m » peut être assimilée à une proposition. A contrario, nous dirons que l'expression « Jean est grand » est ambiguë à moins d'être en mesure de définir précisément la signification d'être grand.

Nous notons différentes propositions par des lettres majuscules en italique {*A*, *B*, *C*, etc.}. Ces propositions seront manipulées à l'aide de la logique usuelle (l'algèbre de Boole).

La notation *AB*, dénommée le *produit logique* ou la *conjonction*, dénote la proposition « les deux propositions *A* et *B* sont vraies ». L'ordre dans lequel nous notons cette expression n'importe pas ; *AB* et *BA* dénotent la même chose.

L'expression *A*+*B*, dénommée la *somme logique* ou la *disjonction*, signifie « au moins une des deux propositions *A* ou *B* est vraie » et est équivalente à l'expression *B*+*A*.

Dans l'algèbre booléenne, le signe = est utilisé pour signifier l'égalité logique, c'est-à-dire l'égalité des valeurs logiques. $A=B$ signifie que la proposition A a la même valeur logique que B. Le symbole ≡ signifie « égal par définition ».

Pour noter des propositions plus complexes nous utiliserons si nécessaire les parenthèses. En l'absence de parenthèses, nous utiliserons les règles de priorité algébrique classiques ; ainsi $AB+C$ dénote l'expression $(AB)+C$, et non pas $A(B+C)$.

Nous utilisons le symbole ¬ pour dénoter la négation d'une proposition : $\neg A \equiv$ « la proposition A est fausse ». Et donc réciproquement : A signifie « $\neg A$ est fausse ». Nous utilisons la convention suivante : $\neg AB$ dénote $(\neg A)B$, « la proposition A est fausse et la proposition B est vraie ». Ainsi $\neg(AB)$ = « la conjonction AB est fausse » ; ce qui est différent de $\neg A \neg B$ = « chacune des deux propositions, A et B, est fausse ».

2.1.2 Variables

Une variable V est associée à un domaine de valeur $D_V = \{v_1, \ldots, v_k\}$, l'ensemble des k valeurs que peut prendre cette variable, sachant que cette variable ne peut prendre qu'une seule valeur à la fois et que cette valeur appartient nécessairement à son domaine de valeur.

Ainsi à chaque variable V peut être associée un ensemble de k propositions logiques : $V_1, V_2, \ldots, V_k$:

$V_1 \equiv [V = v_1]$: « La variable V a pour valeur v_1 »,

$V_2 \equiv [V = v_2]$: « La variable V a pour valeur v_2 »,

…

$V_k \equiv [V = v_k]$: « La variable V a pour valeur v_k ».

Du fait qu'une variable ne peut prendre qu'une valeur et une seule à la fois, les k propositions logiques V_i associées à une variable V sont exhaustives et mutuellement exclusives :

$$\begin{aligned} &V_i\, V_j = \textit{faux} \ \text{ pour } i \neq j \quad \text{(mutuellement exclusives)}, \\ &V_1 + V_2 + \mathtt{L} + V_k = \textit{vrai} \quad \text{(exhaustives)}. \end{aligned} \tag{2.1}$$

Nous noterons différentes variables par des lettres majuscules en caractère droit, complétées selon les cas par des lettres minuscules : {V, X, Y, Z, Dir, Rot …}.

Les informations relatives à une variable V seront notées de la manière suivante :

$$V : D_V, \quad k_V = \text{card}(D_V),$$

où D_V représente le domaine de valeur de la variable V et k_V le nombre de cas possibles.

Par exemple, si la variable Vrot peut varier entre [-10, +10] avec un pas de 1 :

$$\mathtt{Vrot} : D_{Vrot} = \{-10, -9, \ldots, 9, +10\},\ k_{Vrot} = 21.$$

En présence de deux variables V1 et V2 dont les domaines de valeur sont respectivement $D_{V1} = \{v1_1, \ldots, v1_n\}$ et $D_{V2} = \{v2_1, \ldots, v2_m\}$, nous pouvons décrire $n \times m$ propositions logiques élémentaires mutuellement exclusives :

$$V1_1\ V2_1,\ V1_1\ V2_2,\ \ldots,\ V1_1\ V2_m,$$
$$V1_2\ V2_1,\ V1_2\ V2_2,\ \ldots,\ V1_2\ V2_m,$$
$$\ldots$$
$$V1_n\ V2_1,\ V1_n\ V2_2,\ \ldots,\ V1_n\ V2_m.$$

Chacune de ces $n \times m$ propositions peut être interprétée de la manière suivante :

$$V1_i\ V2_j \equiv [\texttt{V1} = \texttt{v1}_\texttt{i}][\texttt{V2} = \texttt{v2}_\texttt{j}] :$$

« La variable `V1` a pour valeur $\texttt{v1}_\texttt{i}$ » **et** « La variable `V2` a pour valeur $\texttt{v2}_\texttt{j}$ ».

Nous résumons l'ensemble des $n \times m$ propositions par la notation `V1 V2`.

De la même manière, nous notons `V1+V2` l'ensemble des $n \times m$ propositions logiques (non mutuellement exclusives) :

$$V1_1+V2_1,\ V1_1+V2_2,\ \ldots,\ V1_1+V2_m,$$
$$V1_2+V2_1,\ V1_2+V2_2,\ \ldots,\ V1_2+V2_m,$$
$$\ldots$$
$$V1_n+V2_1,\ V1_n+V2_2,\ \ldots,\ V1_n+V2_m.$$

Que nous interprétons comme :

$$V1_i+V2_j \equiv [\texttt{V1} = \texttt{v1}_\texttt{i}] + [\texttt{V2} = \texttt{v2}_\texttt{j}] :$$

« La variable `V1` a pour valeur $\texttt{v1}_\texttt{i}$» **ou** « La variable `V2` a pour valeur $\texttt{v2}_\texttt{j}$ ».

Attention : « `V1 V2` » peut être considérée comme une nouvelle variable, par contre « `V1+V2` » n'est pas une nouvelle variable, les $n \times m$ propositions logiques associées à « `V1+V2` » n'étant pas mutuellement exclusives.

2.1.3 Probabilité

La notion de plausibilité intervient lorsque nous ne sommes pas en mesure de connaître de manière certaine la valeur de vérité d'une proposition. Intuitivement, nous définissons la plausibilité comme le degré de certitude que nous avons dans la véracité d'une proposition. Ce degré de certitude est fortement lié à l'ensemble de connaissances dont nous disposons et sur la base desquelles nous souhaitons juger de la véracité d'une proposition.

Nous renvoyons le lecteur aux deux articles ([Bessière98a] [Bessière98b]) insérés en annexe (Annexe A et Annexe B) pour une présentation plus détaillée de la formalisation de la notion de plausibilité. Cette formalisation se concrétise par une théorie du raisonnement plausible, « Probability as Logic », proposée par le physicien E.T. Jaynes [Jaynes98]. Le fondement de cette théorie est le théorème de Cox (voir [Cox46], [Cox61], [Cox79]), qui montre, en partant d'un ensemble de desiderata, que la notion intuitive de plausibilité se formalise par la notion mathématique de probabilité[1].

La probabilité d'une certaine proposition A sera, en général, dépendante du fait qu'une autre proposition B est vraie. Nous adopterons la notation suivante :

---

1 Dans un récent article [Halpern99], la démonstration de Cox de l'unicité de la théorie des probabilités pour la manipulation de la plausibilité a été mise en cause. Néanmoins, cette mise en cause ne concerne en aucun cas la capacité de la théorie des probabilités de manipuler la notion de plausibilité.

$$P(A \mid B) \tag{2.2}$$

pour parler de « la probabilité conditionnelle que A est vraie, sachant que B est vraie », ou encore « A sachant B ». Ainsi, par exemple,

$$P(A \mid B\ C) \tag{2.3}$$

(qui se lit « A sachant B et C ») représente la probabilité que A soit vraie, sachant que les deux propositions B et C sont vraies. Et,

$$P(A+B \mid C\ D) \tag{2.4}$$

représente la probabilité qu'au moins une des deux propositions A et B soit vraie, sachant que les deux propositions C et D sont vraies.

Nous n'écrirons jamais la probabilité d'une proposition A sous la forme $P(A)$. Nous préciserons toujours les connaissances qui nous permettent d'assigner une probabilité à la proposition A.

En effet, nous considérons que la probabilité d'une proposition A ne peut être définie que relativement à une connaissance d'arrière plan (ou connaissance préalable) : C. Nous noterons $P(A \mid C)$: la probabilité que la proposition A soit vraie conditionnellement à toutes les évidences dont nous disposons et que nous résumons dans C. C'est en ce sens que nous parlerons de « probabilité subjective ». Toute probabilité qui n'est conditionnée que par C seule est appelée « probabilité a priori ».

Dans l'intérêt d'éviter des problèmes impossibles, nous ne nous permettrons pas de raisonner à partir de prémisses impossibles ou mutuellement contradictoires. Par conséquent, nous ne chercherons pas à définir $P(A \mid B\ C)$ lorsque B et C sont des propositions mutuellement contradictoires. Quand de telles notations se présenteront, il sera implicitement compris que les propositions B et C sont compatibles.

L'expression $P(\mathtt{X} \mid C)$ dénote la distribution de probabilité attribuée a priori (au vu des connaissances préalables C) à la variable $\mathtt{X}$, c'est-à-dire aux différentes valeurs que peut prendre cette variable. Dans le cas où $\mathtt{X}$ est une variable discrète dont le domaine de valeur est $\mathtt{D_X}$, nous notons $P(\mathtt{X} \mid C)$ l'ensemble des probabilités $P(X_i \mid C)$:

$$\begin{aligned} &\forall \mathtt{x_i} \in \mathtt{D_X}, \\ &P\left(X_i \mid C\right) = P\left(\left[\mathtt{X} = \mathtt{x_i}\right] \mid C\right). \end{aligned} \tag{2.5}$$

2.2 Règles de calcul

Les deux règles fondamentales sur lesquelles repose l'inférence probabiliste sont *la règle du produit* et la *règle de normalisation* ([Cox46], [Cox61], [Cox79]). Nous en déduisons des versions associées pour les variables.

2.2.1 Règles de calcul pour les propositions

Cette section présente les règles de calcul pour la manipulation de propositions élémentaires. La *règle du produit* permet d'exprimer la probabilité conjointe de deux

propositions comme le produit de deux probabilités élémentaires. La *règle de normalisation* est la deuxième règle fondamentale nécessaire à l'inférence. Elle exprime le fait que la somme des probabilités de tous les cas possibles d'un terme vaut 1. De ces deux règles, nous pouvons dériver *la règle de la disjonction* (ou règle de la somme).

2.2.1.1 [R1] : règle du produit

La *règle du produit* **[R1]** donne la probabilité de la conjonction de deux propositions A et B :

$$\mathrm{P}(A\ B \mid C) = \mathrm{P}(A \mid C)\mathrm{P}(B \mid A\ C) = \mathrm{P}(B \mid C)\mathrm{P}(A \mid B\ C). \qquad \textbf{[R1]}$$

Sachant la prémisse C, la probabilité de la conjonction de deux assertions est déterminée par le produit de la probabilité de la première assertion sachant la prémisse C ($\mathrm{P}(A \mid C)$) et de la probabilité de la seconde assertion sachant une prémisse qui est la conjonction de la prémisse donnée C et de la première assertion ($\mathrm{P}(B \mid A\ C)$). La commutativité de la conjonction logique implique que les positions des variables A et B peuvent être interverties à loisir et donc l'existence de deux décompositions possibles.

2.2.1.2 [R2] : règle de normalisation

La *règle de normalisation* **[R2]** exprime la relation entre les probabilités d'une proposition et de sa négation :

$$\mathrm{P}(A \mid C) + \mathrm{P}(\neg A \mid C) = 1. \qquad \textbf{[R2]}$$

La somme de la probabilité d'une assertion A sachant une certaine prémisse C et de la probabilité de sa négation $\neg A$ sachant la prémisse C vaut 1.

2.2.1.3 Règle de la disjonction

Tout comme en logique, où les opérateurs de conjonction et de négation permettent d'exprimer toute expression logique, les deux règles, *règle du produit* **[R1]** et *règle de normalisation* **[R2]**, forment la base à partir de laquelle tous les calculs des probabilités se dérivent. De ces deux règles, nous pouvons dériver *la règle de la disjonction* (ou règle de la somme).

Partant des deux règles **[R1]** et **[R2],** nous pouvons dériver *la règle de la disjonction*, obtenue pour la disjonction de deux propositions :

$$\mathrm{P}(A + B \mid C) = \mathrm{P}(A \mid C) + \mathrm{P}(B \mid C) - \mathrm{P}(A\ B \mid C). \qquad (2.6)$$

En effet, en appliquant la règle **[R1]** et la règle **[R2]** successivement, nous obtenons :

$$
\begin{aligned}
\mathrm{P}(A+B \mid C) &= 1-\mathrm{P}(\neg A\ \neg B \mid C) \\
&= 1-\mathrm{P}(\neg A \mid C)\mathrm{P}(\neg B \mid \neg A\ C) \\
&= 1-\mathrm{P}(\neg A \mid C)\left[1-\mathrm{P}(B \mid \neg A\ C)\right] \\
&= \mathrm{P}(A \mid C)+\mathrm{P}(\neg A\ B \mid C) \\
&= \mathrm{P}(A \mid C)+\left(\mathrm{P}(B \mid C)\mathrm{P}(\neg A \mid B\ C)\right) \\
&= \mathrm{P}(A \mid C)+\mathrm{P}(B \mid C)\left[1-\mathrm{P}(A \mid B\ C)\right] \\
&= \mathrm{P}(A \mid C)+\mathrm{P}(B \mid C)-\mathrm{P}(AB \mid C).
\end{aligned}
\tag{2.7}
$$

Si les propositions A et B sont mutuellement exclusives, de l'équation (2.7) nous obtenons la relation suivante :

$$\mathrm{P}(A+B \mid C)=\mathrm{P}(A \mid C)+\mathrm{P}(B \mid C). \qquad \textbf{[T1]}$$

2.2.2 Règles de calcul pour les variables

2.2.2.1 Règle du produit pour les variables

Nous notons P(X Y | C) l'ensemble des $n \times m$ probabilités conjointes P(X_i Y_j | C).

Nous écrivons la relation de dépendance reliant deux variables X et Y sachant une certaine prémisse C par : P(X Y | C). Dans ce cadre, la règle du produit **[R1]** s'écrira comme :

$$\mathrm{P}(\mathrm{X}\ \mathrm{Y} \mid C)=\mathrm{P}(\mathrm{X} \mid C)\mathrm{P}(\mathrm{Y} \mid \mathrm{X}\ C)=\mathrm{P}(\mathrm{Y} \mid C)\mathrm{P}(\mathrm{X} \mid \mathrm{Y}\ C), \tag{2.8}$$

qui n'est autre qu'un raccourci d'écriture pour :

$$
\begin{aligned}
&\forall \mathrm{x}_i \in \mathrm{D}_{\mathrm{X}}, \forall \mathrm{y}_j \in \mathrm{D}_{\mathrm{Y}} \\
\mathrm{P}\left(\left[\mathrm{X}=\mathrm{x}_i\right]\left[\mathrm{Y}=\mathrm{y}_j\right] \mid C\right) &= \mathrm{P}\left(\left[\mathrm{X}=\mathrm{x}_i\right] \mid C\right)\mathrm{P}\left(\left[\mathrm{Y}=\mathrm{y}_j\right] \mid \left[\mathrm{X}=\mathrm{x}_i\right] C\right) \\
&= \mathrm{P}\left(\left[\mathrm{Y}=\mathrm{y}_j\right] \mid C\right)\mathrm{P}\left(\left[\mathrm{X}=\mathrm{x}_i\right] \mid \left[\mathrm{Y}=\mathrm{y}_j\right] C\right),
\end{aligned}
\tag{2.9}
$$

ou encore :

$$
\begin{aligned}
&\forall \mathrm{x}_i \in \mathrm{D}_{\mathrm{X}}, \forall \mathrm{y}_j \in \mathrm{D}_{\mathrm{Y}}, \text{ sachant que } X_i \equiv \left[\mathrm{X}=\mathrm{x}_i\right] \text{ et } Y_j \equiv \left[\mathrm{Y}=\mathrm{y}_j\right], \\
&\mathrm{P}(X_i\ Y_j \mid C)=\mathrm{P}(X_i \mid C)\mathrm{P}(Y_j \mid X_i\ C)=\mathrm{P}(Y_j \mid C)\mathrm{P}(X_i \mid Y_j\ C).
\end{aligned}
\tag{2.10}
$$

2.2.2.2 Règles de normalisation pour les variables

La règle de normalisation exprime le fait que la somme des probabilité de tous les cas possibles d'une variable vaut 1.

Soient la variable X **:**

$$\mathrm{X} : \mathrm{D}_{\mathbf{x}}, k_{\mathbf{x}}.$$

Sachant que $X_i \equiv [\mathtt{X} = \mathtt{x}_i]$:

$$
\begin{aligned}
&P(X_1 \mid C) + P(\neg X_1 \mid C) = 1 \quad \text{(par } \textbf{[R2]}\text{)} \\
&P(X_1 \mid C) + P(X_2 + \mathtt{L} + X_n \mid C) = 1 \quad (X_i \text{ exhaustives} : \neg X_1 = X_2 + \mathtt{L} + X_n) \\
&P(X_1 \mid C) + P\big(X_2 + (X_3 + \mathtt{L} + X_n) \mid C\big) = 1 \\
&P(X_1 \mid C) + P(X_2 \mid C) + P(X_3 + \mathtt{L} + X_n \mid C) - P\big(X_2(X_3 + \mathtt{L} + X_n) \mid C\big) = 1 \quad \text{(par } \textbf{[T1]}\text{)} \\
&P(X_1 \mid C) + P(X_2 \mid C) + P(X_3 + \mathtt{L} + X_n \mid C) = 1 \quad (X_i \text{ mutuellement exclusive}) \\
&\mathtt{M} \\
&P(X_1 \mid C) + P(X_2 \mid C) + \mathtt{L}\ P(X_n \mid C) = 1 \\
&\sum_{\mathtt{x}_i \in D_{\mathtt{X}}} P(X_i \mid C) = 1.
\end{aligned}
\tag{2.11}
$$

Nous résumons cette relation par la règle de normalisation notée de la manière suivante :

$$\sum_{\mathtt{X}} P(\mathtt{X} \mid C) = 1. \tag{2.12}$$

2.2.2.3 Règle de la disjonction pour les variables

Soient deux variables X et Y ayant pour domaines respectivement D_X et D_Y ; la règle de la disjonction s'exprime ainsi :

$$P(\mathtt{X} + \mathtt{Y} \mid C) = P(\mathtt{X} \mid C) + P(\mathtt{Y} \mid C) - P(\mathtt{XY} \mid C). \tag{2.13}$$

Cette écriture résume la relation suivante :

$$
\begin{aligned}
&\forall \mathtt{x}_i \in D_{\mathtt{X}}, \forall \mathtt{y}_j \in D_{\mathtt{Y}}, \text{ sachant que } X_i \equiv [\mathtt{X} = \mathtt{x}_i] \text{ et } Y_j \equiv [\mathtt{Y} = \mathtt{y}_j], \\
&P(X_i + Y_j \mid C) = P(X_i \mid C) + P(Y_j \mid C) - P(X_i Y_j \mid C).
\end{aligned}
\tag{2.14}
$$

2.2.2.4 Règle de marginalisation

Soient 2 variables X Y, la règle de marginalisation est :

$$\sum_{\mathtt{X}} P(\mathtt{X}\ \mathtt{Y} \mid C) = P(\mathtt{Y} \mid C). \tag{2.15}$$

Cette règle s'obtient par l'application de la règle de normalisation et de la règle du produit :

$$
\begin{aligned}
\sum_{\mathtt{X}} P(\mathtt{X}\ \mathtt{Y} \mid C) &= \sum_{\mathtt{X}} P(\mathtt{Y} \mid C) P(\mathtt{X} \mid \mathtt{Y}\ C) \\
&= P(\mathtt{Y} \mid C) \sum_{\mathtt{X}} P(\mathtt{X} \mid \mathtt{Y}\ C) \\
&= P(\mathtt{Y} \mid C).
\end{aligned}
\tag{2.16}
$$

2.2.3 Résumé

Nous résumons ci-dessous les trois règles dont nous aurons besoin par la suite :

Règle du produit :

$$P(\mathtt{X}\ \mathtt{Y} \mid C) = P(\mathtt{X} \mid C)P(\mathtt{Y} \mid \mathtt{X}\ C) = P(\mathtt{Y} \mid C)P(\mathtt{X} \mid \mathtt{Y}\ C). \tag{2.17}$$

Règle de normalisation :

$$\sum_{\mathtt{X}} P(\mathtt{X} \mid C) = 1. \tag{2.18}$$

Règle de marginalisation :

$$\sum_{\mathtt{X}} P(\mathtt{X}\ \mathtt{Y} \mid C) = P(\mathtt{Y} \mid C). \tag{2.19}$$

2.3 Méthode

Ce paragraphe présente les principes de la méthode que nous allons suivre pour « programmer » le Khepera tout au long des différentes expériences décrites par la suite. Nous allons construire un objet formel qui sera la brique de base de nos programmes : la *description*. La programmation d'un robot à l'aide d'une description nécessite trois phases :

- Spécification des connaissances préalables *C* ;
- Identification des valeurs des paramètres des distributions de probabilité ;
- Utilisation de la description.

Nous allons maintenant définir le concept de *description* et décrire plus précisément chacune de ces trois phases.

2.3.1 Description (définition)

La *description* est la notion centrale de notre système de programmation probabiliste. C'est par son biais que le programmeur décrit les connaissances préalables dont il dispose (phase de spécification), soumet ces connaissances à l'expérience (phase d'identification) et enfin les restitue sous la forme de *questions probabilistes* (phase d'utilisation).

Une description est dénotée formellement par la probabilité conjointe d'un ensemble de variables `V1` … `Vn` déterminée au vu des connaissances préalables spécifiées par le programmeur (notées *C*) et d'un ensemble de données expérimentales *D* :

$$\begin{gathered} P(\mathtt{V1...Vn} \mid D\ C), \\ \mathtt{V1...Vn} : \text{ ensemble des variables}, \\ D : \text{ données expérimentales}, \\ C : \text{ connaissances préalables}. \end{gathered} \tag{2.20}$$

Ainsi, l'ensemble des connaissances devant être fourni se trouve circonscrit par les connaissances préalables *C* et le jeu de données expérimentales *D*.

2.3.2 Spécification

La phase de spécification est la partie la plus délicate du travail du programmeur. Au cours de cette phase, le programmeur énonce clairement et *explicitement* ses connaissances et celles qui résultent d'un processus adaptatif dépendant d'un jeu particulier de données expérimentales. Ces connaissances se divisent en trois :

- Le choix des variables pertinentes.
- L'expression de la dépendance entre les différentes variables retenues sous la forme d'un produit de distributions de probabilité élémentaires.
- La forme paramétrique associée à chacune de ces distributions élémentaires.

2.3.2.1 Connaissances préalables structurelles : le choix des variables pertinentes

Dans l'expression des connaissances fournies par le programmeur, la première étape consiste à choisir un ensemble de variables sur lequel va porter la description.

Nous appelons *espace des variables* l'ensemble des variables disponibles pour programmer notre robot. Ces variables sont les termes de base avec lesquels notre robot va être en mesure de caractériser et de contrôler son état. À ce point, nous pouvons distinguer trois types de variables : les variables sensorielles (notées S), les variables motrices (notées M) et les variables internes (notées I).

Nous appelons *connaissances préalables structurelles* les connaissances permettant de définir l'ensemble des variables `V1` … `Vn` pour la description, et permettant de spécifier pour chacune d'elles son domaine de valeur et le nombre d'états possibles. Toutes les autres variables de l'espace des variables sont ainsi supposées non pertinentes pour le problème considéré.

Les variables sensorielles **S**

Nous qualifions de variables sensorielles celles qui permettent de fournir de l'information au robot. Par les valeurs que prennent ces variables, le robot est en mesure de caractériser son état, c'est-à-dire distinguer les situations dans lesquelles il se trouve.

Cette information peut provenir de différentes sources, comprenant les capteurs dont dispose le robot physique, mais aussi le résultat de pré-traitements spécifiques appliqués aux valeurs fournies par les capteurs, les consignes permettant à un opérateur d'interagir avec le robot, …, et, plus généralement, toutes les variables sur lesquelles le robot n'a pas d'action directe ; il ne commande pas directement leurs valeurs, elles lui sont imposées par un mécanisme « extérieur ».

Les variables motrices **M**

À l'inverse des variables sensorielles, ces variables sont celles dont les valeurs peuvent (doivent) être changées par le robot, ce qui à pour conséquence de modifier l'état physique de ses actionneurs.

Les variables internes **I**

Ces variables représentent des ressources internes, c'est-à-dire qu'elles permettent au robot de coder des états internes. Le robot peut lire et modifier leurs valeurs. Elles sont utilisées comme intermédiaires de calcul, comme ressources pour structurer un programme et comme

moyen de mémoriser et d'intégrer des informations. Ces variables n'auront un sens que mises en relation avec des variables sensorielles ou motrices.

Piloter un robot va consister à déterminer les valeurs des variables motrices en prenant en compte les valeurs des variables sensorielles et, éventuellement, les valeurs des variables internes.

2.3.2.2 Connaissances préalables de dépendance : le choix d'une décomposition de la distribution conjointe

La dépendance entre les variables choisies est dénotée formellement par leur probabilité conjointe. La *structure de dépendance* permet de décrire la probabilité conjointe sous la forme d'une décomposition en produit de distributions plus simples.

Supposons que nous ayons retenu 3 variables pour une description : `V1 V2 V3`. Les connaissances préalables de dépendances vont nous permettre de structurer la dépendance entre ces variables. En appliquant la règle du produit nous pouvons écrire la probabilité conjointe de ces trois variables, de treize manières différentes :

$$
\begin{aligned}
P(\texttt{V1 V2 V3} \mid C) &= P(\texttt{V1} \mid C)\, P(\texttt{V2 V3} \mid \texttt{V1}\ C) \\
&= P(\texttt{V2} \mid C)\, P(\texttt{V1 V3} \mid \texttt{V2}\ C) \\
&= P(\texttt{V3} \mid C)\, P(\texttt{V1 V2} \mid \texttt{V3}\ C) \\
&= P(\texttt{V1 V2} \mid C)\, P(\texttt{V3} \mid \texttt{V1 V2}\ C) \\
&= P(\texttt{V1} \mid C)\, P(\texttt{V2} \mid \texttt{V1}\ C)\, P(\texttt{V3} \mid \texttt{V1 V2}\ C) \\
&= P(\texttt{V2} \mid C)\, P(\texttt{V1} \mid \texttt{V2}\ C)\, P(\texttt{V3} \mid \texttt{V1 V2}\ C) \\
&= P(\texttt{V1 V3} \mid C)\, P(\texttt{V2} \mid \texttt{V1 V3}\ C) \\
&= P(\texttt{V1} \mid C)\, P(\texttt{V3} \mid \texttt{V1}\ C)\, P(\texttt{V2} \mid \texttt{V1 V3}\ C) \\
&= P(\texttt{V3} \mid C)\, P(\texttt{V1} \mid \texttt{V3}\ C)\, P(\texttt{V2} \mid \texttt{V1 V3}\ C) \\
&= P(\texttt{V2 V3} \mid C)\, P(\texttt{V1} \mid \texttt{V2 V3}\ C) \\
&= P(\texttt{V2} \mid C)\, P(\texttt{V3} \mid \texttt{V2}\ C)\, P(\texttt{V1} \mid \texttt{V3 V2}\ C) \\
&= P(\texttt{V3} \mid C)\, P(\texttt{V2} \mid \texttt{V3}\ C)\, P(\texttt{V1} \mid \texttt{V2 V3}\ C).
\end{aligned}
$$

Du point de vue des règles de manipulation, ces treize écritures sont équivalentes. En exprimant la probabilité conjointe sous la forme d'une décomposition particulière, nous décrivons une certaine forme de connaissance ou, autrement dit, un certain choix pour décrire la dépendance entre les variables considérées. Ce choix peut être motivé par différentes considérations comme le choix du modèle paramétrique associé à chaque distribution élémentaire, mais aussi la possibilité d'exprimer des hypothèses d'indépendance conditionnelle. Par exemple, nous pouvons exprimer l'hypothèse suivante : à la condition de connaître la valeur de `V3`, la valeur de la variable `V1` ne dépend pas de `V2` ; et ainsi « éliminer » la variable `V2` dans l'expression de $P(\texttt{V1} \mid \texttt{V2 V3}\ C)$:

$$P(\texttt{V1} \mid \cancel{\texttt{V2}}\ \texttt{V3}\ C) = P(\texttt{V1} \mid \texttt{V3}\ C).$$

Nous appelons l'expression de ces choix les *connaissances préalables de dépendances*. Ces dernières permettent d'exprimer, sans équivoque, la description comme un produit de distributions élémentaires et d'énoncer des considérations d'indépendances conditionnelles entre les variables retenues.

2.3.2.3 Connaissances préalables d'observation : le choix des formes paramétriques

Nous venons de décrire les connaissances permettant de spécifier les variables manipulées et la décomposition de la distribution conjointe en un produit de distributions élémentaires. Afin de rendre effective notre description, il reste à préciser les valeurs associées à chacune d'elles. C'est le rôle des *connaissances préalables d'observation*.

Par l'expression de ces connaissances, nous allons fournir des a priori sur les valeurs des distributions de probabilité et la manière dont ces valeurs seront modifiées par l'expérience.

Ce sont ces connaissances qui vont nous permettre de décrire les formes paramétriques associées à chaque distribution élémentaire, la valeur de leurs paramètres en l'absence de données expérimentales et les mécanismes de modification de ces derniers sous l'influence de ces données D (pouvant se résumer à ne rien changer pour les distributions supposées ne pas évoluer en fonction des données).

Dans les différentes expériences proposées, nous nous limiterons à l'utilisation des trois modèles paramétriques suivants :

Loi Uniforme

C'est une forme paramétrique dont le seul paramètre est le nombre de cas de la variable. Cette distribution n'est pas influencée par les données expérimentales. Elle indique que la variable peut prendre indifféremment n'importe quelle valeur dans son domaine.

Notation **U**.

$$\begin{aligned} &\mathrm{P}(\mathrm{V} \mid D\ C) = \mathrm{P}(\mathrm{V} \mid C) = \mathbf{U}_{k\mathrm{V}}(\mathrm{V}) \\ &\forall \mathrm{v}_i \in \mathrm{D_V}, \\ &\mathrm{P}\left(\left[\mathrm{V} = \mathrm{v}_i\right] \mid C\right) = \frac{1}{k_\mathrm{V}}. \end{aligned} \tag{2.21}$$

Loi de succession de Laplace

Cette forme admet autant de paramètres que le nombre de cas de la variable sur laquelle elle porte ; ces paramètres se calculent aisément au vu des données expérimentales. Pour un grand nombre de données expérimentales, la distribution résultant de l'identification peut être assimilée à une normalisation du nombre d'occurrences de chaque cas. En absence de données expérimentales, nous obtenons une distribution uniforme.

Notation **L**

$$P(\mathtt{V} \mid D\ C) = \mathbf{L}_{n_1,\, n_2,\, \ldots,\, n_{k\mathtt{V}}}(\mathtt{V})$$
$$P\left(\left[\mathtt{V} = \mathtt{v}_i\right] \mid D\ C\right) = \frac{n_i + 1}{N + k_{\mathtt{V}}} \qquad (2.22)$$

n_i : nombre de fois où V = vi dans l' ensemble des données expérimentales D,

N : le nombre total de données expérimentales, $N = \sum_{i=1}^{k\mathtt{V}} n_i$.

Loi de Gauss

Cette forme paramétrique classique (appelée également loi Normale) correspond au choix de ne retenir des données expérimentales que la moyenne des valeurs rencontrées et leur variance.

Notation **G**

$$P(\mathtt{V} \mid D\ C) = \mathbf{G}_{\mu,\, \sigma}(\mathtt{V})$$
$$P\left(\left[\mathtt{V} = \mathtt{vi}\right] \mid D\ C\right) = \int_{\mathtt{vi}-\varepsilon}^{\mathtt{vi}+\varepsilon} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(v-\mu)^2}{2\sigma^2}}\, dv \qquad (2.23)^{2}$$

μ : la moyenne de V pour l' ensemble des données expérimentales D

σ : l' écart type de V pour l' ensemble des données expérimentales D

Nous utiliserons également des distributions de probabilité représentées sous formes discrètes ou fonctionnelles. Ces formes seront décrites spécifiquement dans le cadre de chaque expérience.

2.3.2.4 Connaissances préalables (définition)

Les trois types de connaissances que le programmeur doit spécifier ne sont pas indépendantes mais au contraire fortement tributaires les unes des autres. L'ensemble de ces trois connaissances forme ce que nous définissons comme étant les *connaissances préalables* notées C :

Les connaissances préalables C sont donc :

- *Les connaissances préalables structurelles* qui définissent l'ensemble des variables V1 … Vn retenues. Leurs interdépendances seront exprimées sous forme probabiliste par la description.
- *Les connaissances préalables de dépendance* (ou structure de dépendance) qui décomposent sans équivoque la distribution conjointe en un produit de distributions élémentaires. Ici s'expriment des considérations d'indépendance conditionnelle entre les variables.
- *Les connaissances préalables d'observation* (ou structure paramétrique) qui décrivent les formes paramétriques associées à chaque distribution élémentaire, la valeur de leurs paramètres en absence de données et les mécanismes de modification de ces derniers sous l'influence des données expérimentales D.

---

2 Par exemple, dans le cas d'un pas de discrétisation de 1, la valeur ε est choisie à 0.5.

2.3.3 Identification

Une fois les connaissances préalables spécifiées, il reste à fixer les valeurs des formes paramétriques associées à chaque distribution élémentaire de la décomposition. C'est le rôle de la phase d'identification.

Les valeurs des paramètres sont soit données a priori, soit identifiées sur la base d'un jeu de données expérimentales provenant de l'observation d'un même phénomène. De ce jeu de données, un certain nombre d'observables sont évaluées (par exemple la moyenne empirique des moments d'ordre 1 et 2 d'une variable). Sur la base de ces observations, les paramètres des distributions spécifiées par les connaissances préalables sont identifiés (par exemple valeur moyenne et écart-type).

2.3.4 Utilisation

Au terme des phases de spécification et d'identification, nous disposons d'une description complètement définie. La phase d'utilisation va consister à mettre en œuvre les descriptions par le biais de questions probabilistes.

2.3.4.1 Question (définition)

Poser une question consiste à chercher la distribution de probabilité d'un certain nombre de variables de la description connaissant les valeurs d'autres variables.

Si nous notons $\mathtt{E}$ l'ensemble des variables associées à une description, nous appelons respectivement $\mathtt{E}_q$, $\mathtt{E}_c$ et $\mathtt{E}_i$ les trois sous ensembles qui forment une partition de $\mathtt{E}$:

- $\mathtt{E}_q$: le sous-ensemble des variables de $\mathtt{E}$ dont on cherche la distribution ;
- $\mathtt{E}_c$: le sous-ensemble des variables de $\mathtt{E}$ dont on connaît les valeurs ;
- $\mathtt{E}_i$: le sous-ensemble des variables de $\mathtt{E}$ dont on ignore les valeurs.

Nous définissons une question probabiliste comme n'importe quelle expression de la forme :

$$P\left(\mathtt{Vk}...\mathtt{Vl} \mid \left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] D\ C\right) \tag{2.24}$$

$\{\mathtt{Vk}...\mathtt{Vl}\} = \mathtt{E}_q$; ensemble des variables dont on cherche la distribution conjointe ;

$\{\mathtt{Vm}...\mathtt{Vn}\} = \mathtt{E}_c$; ensemble des variables dont on connaît la valeur.

2.3.4.2 Inférence

La connaissance de la distribution conjointe $P(\mathtt{V1}\ldots\mathtt{Vn} \mid D\ C)$ et l'application des règles du produit et de marginalisation nous permet de répondre à toute question de la forme (2.24). Tout d'abord en appliquant la règle du produit, toute question peut s'écrire sous la forme suivante :

$$\mathrm{P}\left(\mathtt{Vk...Vl} \mid \left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] D\ C\right)$$
$$= \frac{\mathrm{P}\left(\mathtt{Vk...Vl}\ \left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] \mid D\ C\right)}{\mathrm{P}\left(\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] \mid D\ C\right)} \tag{2.25}$$

$\{\mathtt{Vk...Vl}\} = \mathtt{E}_q$; ensemble des variables dont on cherche la distribution conjointe ;

$\{\mathtt{Vm...Vn}\} = \mathtt{E}_c$; ensemble des variables dont on connaît la valeur.

En appliquant la règle de marginalisation, nous pouvons exprimer le dénominateur et le numérateur de ce quotient en fonction de la distribution conjointe globale que l'on sait calculer :

$$\frac{\mathrm{P}\left(\mathtt{Vk...Vl}\ \left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] \mid D\ C\right)}{\mathrm{P}\left(\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] \mid D\ C\right)}$$
$$= \frac{\sum_{\mathtt{Vo...Vp}} \mathrm{P}\left(\mathtt{Vk...Vl}\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right]\ \mathtt{Vo...Vp} \mid D\ C\right)}{\sum_{\substack{\mathtt{Vk...Vl} \\ \mathtt{Vo...Vp}}} \mathrm{P}\left(\mathtt{Vk...Vl}\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right]\ \mathtt{Vo...Vp} \mid D\ C\right)} \tag{2.26}$$

$\{\mathtt{Vk...Vl}\} = \mathtt{E}_q$; ensemble des variables dont on cherche la distribution conjointe ;

$\{\mathtt{Vm...Vn}\} = \mathtt{E}_c$; ensemble des variables dont on connaît la valeur ;

$\{\mathtt{Vo...Vp}\} = \mathtt{E}_i$; ensemble des variables dont on ignore la valeur.

Le dénominateur de cette expression est obtenue par la marginalisation du numérateur sur les variables recherchées :

$$\sum_{\substack{\mathtt{Vk...Vl} \\ \mathtt{Vo...Vp}}} \mathrm{P}\left(\mathtt{Vk...Vl}\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right]\ \mathtt{Vo...Vp} \mid D\,C\right)$$
$$= \sum_{\mathtt{Vk...Vl}} \left(\sum_{\mathtt{Vo...Vp}} \mathrm{P}\left(\mathtt{Vk...Vl}\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right]\ \mathtt{Vo...Vp} \mid D\,C\right)\right). \tag{2.27}$$

Le dénominateur est une expression qui ne dépend que des valeurs particulières des variables connues. Il joue le rôle d'une constante de normalisation que nous noterons généralement Z. Ainsi, la résolution d'une question quelconque s'exprime sous la forme suivante :

$$\frac{\mathrm{P}\left(\mathtt{Vk...Vl}\ \left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] \mid D\ C\right)}{\mathrm{P}\left(\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right] \mid D\ C\right)}$$
$$= \frac{1}{Z} \sum_{\mathtt{Vo...Vp}} \mathrm{P}\left(\mathtt{Vk...Vl}\left[\mathtt{Vm} = \mathtt{vm}_i\right]...\left[\mathtt{Vn} = \mathtt{vn}_j\right]\ \mathtt{Vo...Vp} \mid D\ C\right). \tag{2.28}$$

Cette méthode de résolution d'une question est purement symbolique et ne tient pas compte des simplifications que peuvent faire apparaître les connaissances préalables fournies lors de la phase de spécification ainsi que les modes de décision que l'on veut appliquer par la suite.

2.3.4.3 Décision

Le résultat de l'inférence fournit une distribution de probabilité sur les variables recherchées. Rendre effectif le résultat d'une question va consister à choisir une valeur particulière pour chacune de ces variables.

Différentes stratégies peuvent être mises en œuvre, comme le choix de la valeur la plus probable, ou des heuristiques de décision plus sophistiquées pour lesquelles on adjoint à l'état de connaissance probabiliste des fonctions d'utilité (ou de coût). Ces fonctions représentent, sous forme de contraintes, l'intérêt ou le danger d'une décision particulière. Pour de telles heuristiques, choisir une valeur particulière se ramène à résoudre un problème d'optimisation sous contraintes.

Dans le cadre des expériences que nous présentons, nous nous sommes limité à une heuristique de décision très simple qui consiste à obtenir la valeur des variables recherchées par un tirage selon la distribution de probabilité obtenue.

2.4 Programme et interpréteur probabiliste

Nous pouvons maintenant définir ce que nous nous appelons : un programme et un interpréteur probabiliste.

2.4.1 Programme probabiliste (définition)

Pour nous, la définition d'un programme probabiliste est donnée par la spécification de trois composantes : la description, la question et la décision.

> Programme probabiliste :
> description + question + décision

La description spécifie formellement l'ensemble des choix retenus pour décrire un phénomène. Elle permet de définir sans ambiguïté la distribution conjointe sur un ensemble de variables. Rappelons rapidement quels sont ces choix :

- Le choix des variables retenues ;
- Le choix de la décomposition ;
- Le choix des formes paramétriques ;
- Le choix des données ;
- Le choix des méthodes d'identification.

Nous désignons l'ensemble de ces choix par un identificateur rappelant les connaissances préalables utilisées pour les définir, par exemple : *Cp_tâche_reflexe* .

La question définit formellement une partition de l'ensemble E en trois sous-ensembles : E_q, E_c et E_i. Elle exprime une distribution de probabilité conditionnelle. Cette distribution est formellement calculable à partir d'une marginalisation de la distribution conjointe décrite par la description.

La décision permet d'affecter une valeur aux variables de l'ensemble E_q en fonction de la distribution définie par la question. Ces valeurs sont le résultat du programme.

Dans ce document, nous allons définir plusieurs programmes probabilistes pour la robotique. Ces programmes seront toujours présentés dans un encadré spécifiant les trois composants dont nous venons d'évoquer. La spécificité « robotique » de ces programmes vient du choix généralement fait pour les ensembles E_q et E_c :

- E_q sera généralement l'ensemble des variables motrice **M**
- E_c l'ensemble des variables sensorielles **S**.

Un programme sera représenté comme suit :

1/ Description utilisée : nous indiquerons ici la (ou les) description(s) nécessaire(s) pour la tâche :

Cp_tâche_reflexe

2/ Question : nous rappellerons pour la question associée à la tâche, l'ensemble E_q des variables recherchées, l'ensembles E_c des variables connues, l'ensemble E_i des variables ignorées et la question correspondante posée à la description :

E_q : {M}

E_c : {S}

E_i : Ø

Question : P(M | S *Cp_tâche_reflexe*)

3/ Décision : nous préciserons le type de décision appliquée :

Tirage aléatoire

2.4.2 Interpréteur probabiliste (définition)

Un interpréteur probabiliste est un programme informatique qui accepte un programme probabiliste en entrée et fournit un ensemble de valeur en sortie pour les variables E_q. La construction d'un interpréteur est un sujet de recherche à part entière. Deux difficultés majeures doivent être abordées ; l'une concerne les problèmes de marginalisation et l'autre les problèmes de tirages aléatoires. Ces deux problèmes ne reçoivent pas en général de réponses simples dans les espaces de grandes dimensions et de nombreuses techniques ont été développées pour tenter de les résoudre (réseaux bayésien [Lauritzen88] [Pearl91] [Shachter86], méthodes de Monte Carlo [MacKay96] [Neal93] [Mekhnacha99]).

Il a été développé dans l'équipe LAPLACE un interpréteur général que nous appelons « moteur d'inférence probabiliste ». Ce moteur a pour ambition d'être capable d'interpréter tout programme probabiliste donné sous la forme précédemment expliquée. Il peut être comparé aux nombreux moteurs d'inférences logiques qui existent et permettent d'automatiser le raisonnement symbolique (voir, par exemple, Prolog [Colmerauer86], LogLisp [Robinson83] ou Lolita [Bessière87]). Pour une question donnée, l'expression correspondante est tout d'abord simplifiée symboliquement par le moteur. Sa valeur

numérique est ensuite estimée. La complexité du calcul numérique vient évidemment des sommes. Ce moteur intègre une méthode d'estimation performante de ces sommes qui fait notamment appel à un algorithme d'optimisation de type « génétique » et à une représentation des distributions sous forme très compacte et efficace, inspirée des « Octree » ([Jung97]). Cet interpréteur à été utilisé pour notre programme.

Dans les cas non triviaux, nous avons choisi d'expliciter la partie formelle de la résolution d'une question. Par ailleurs, la taille des espaces considérés sera en général suffisamment petite pour nous permettre d'utiliser des méthodes explicites de sommation et de tirages aléatoires. Sans perte de généralité, nous avons donc engendré un interpréteur spécifique à chaque programme probabiliste. La spécificité « robotique » de cet interpréteur réside dans le contexte informatique dans lequel il est utilisé. Comme nous le verrons au chapitre suivant, cet interpréteur est appelé tous les dixièmes de seconde avec le même programme pour calculer les valeurs qui seront appliquées aux actionneurs du robot.

Chapitre 3

Présentation de la plate-forme expérimentale

Ce chapitre décrit la plate-forme expérimentale utilisée au cours des expérimentations. Le dispositif (voir Figure 3-1) est constitué d'un petit robot mobile, le Khepera, évoluant dans un environnement de 1 m^2. Le pilotage du robot est déporté sur un ordinateur hôte de type Macintosh©. Il se présente sous la forme d'une boucle de commande synchronisée sur l'arrivée des données sensorielles. La communication entre l'ordinateur et le robot s'effectue par le biais d'une liaison série de type RS232. De plus nous disposons d'un manche à balai afin de télé-opérer le robot.

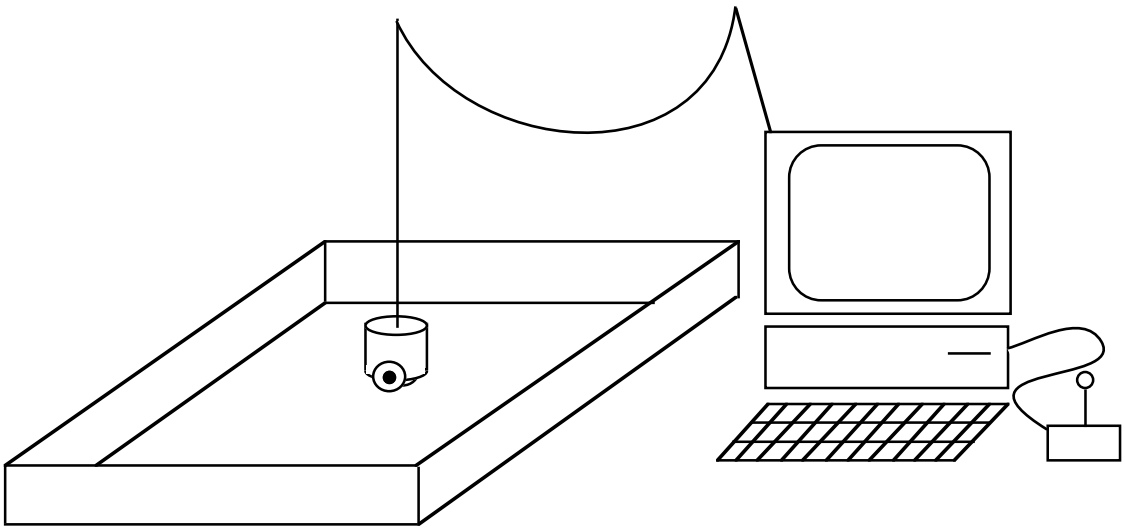


Figure 3-1 — Schéma de la plate-forme expérimentale.

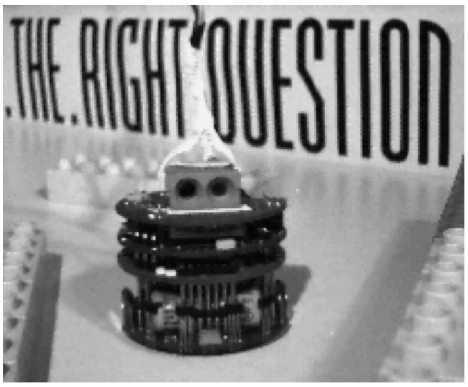


Figure 3-2 — Le Khepera en action.

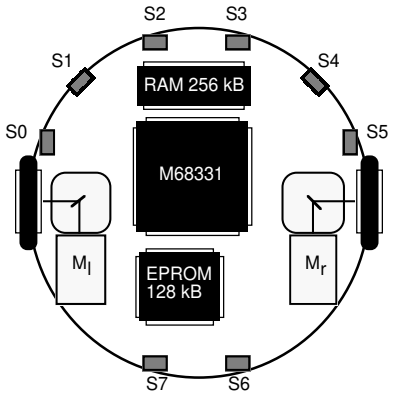


Figure 3-3 — Schéma du Khepera

3.1 Le Khepera

Les différentes expériences présentées ont été réalisées avec un robot mobile miniature, développé à l'École Polytechnique Fédérale de Lausanne (EPFL) : le Khepera (voir [Mondada93], [Mondada94]).

Le Khepera (diamètre 55 mm, hauteur 30 mm, poids 80 g dans sa configuration de base) est montré sur la Figure 3-2 et schématisé sur la Figure 3-3. Bien que de dimension réduite, il dispose d'un processeur embarqué performant (M68331, 32 bits, cadencé à 16 MHz), associé à une EEPROM de 128 K octets et une RAM statique de 256 Koctets. Le BIOS (Basic I/Os System) est le système bas niveau embarqué dans le robot. Il offre des capacités multi-tâches et permet la gestion de plusieurs modules logiciels : acquisition et conversion des données sensorielles, asservissements des moteurs, contrôle de la communication entre différents modules du robot et l'extérieur.

La partie motrice est constituée de deux roues latérales commandées indépendamment. Elles offrent en particulier au Khepera la possibilité de tourner sur place. Chaque roue est entraînée par un moteur continu (M_l et M_r) associé à un réducteur à vis sans fin et à un codeur incrémental d'une résolution de 600 impulsions par tour de roue (~ 12 pulses/mm parcouru par le robot). Ces capteurs (`PosG` et `PosD`) fournissent une information sur l'égo-mouvement du robot. À chaque moteur sont associés deux asservissements classiques de type P.I.D. permettant le contrôle du robot en vitesse ou en position.

Les ressources système du robot fournissent des ordres permettant de spécifier les valeurs des vitesses de rotation des moteurs gauche et droit (`Vg` et `Vd`). L'unité est le pulse/10ms, ce qui correspond à 8 mm par seconde. Nous avons préféré travailler avec des variables homogènes aux vitesses de rotation (`Vrot`) et de translation (`Vtrans`) du robot que nous exprimons dans la même unité (pulse/10ms). Ces 4 variables sont reliées les unes aux autres par les relations :

$$\begin{aligned} \texttt{Vg} &= \texttt{Vtrans} + \texttt{Vrot} \\ \texttt{Vd} &= \texttt{Vtrans} - \texttt{Vrot}. \end{aligned} \tag{3.1}$$

Les tableaux suivants (Tableau 3.1 et Tableau 3.2) indiquent la correspondance entre les commandes Vrot et Vtrans et les ordres de grandeur en mm/sec pour la vitesse de translation, en °/sec pour la vitesse de rotation :

Vtrans	$mm.s^{-1}$
1	8
5	40
10	80

Tableau 3.1 — Commande en vitesse de translation.

Vrot	$tour.s^{-1}$	$°.s^{-1}$
1	0.05	18
5	0.25	90
10	0.5	180

Tableau 3.2 — Commande en vitesse de rotation.

Le Khepera dispose d'un anneau de huit capteurs infrarouges (S_0 à S_7). Chaque capteur est composé d'un émetteur de lumière infrarouge et d'un récepteur. Ces capteurs permettent deux types de mesures : une de la luminosité ambiante en mode récepteur (`Lm0` à `Lm7`) et l'autre de la réflexion de la lumière par les obstacles en utilisant le mode émetteur et récepteur (`Px0` à `Px7`). Cette dernière mesure permet d'obtenir une information relative à la distance des obstacles pouvant varier entre 1 et 5 cm, dépendant fortement de leur nature (orientation, couleur, matière, etc.) et des conditions de luminosité ambiantes.

Ces ressources sont enrichies par l'adjonction d'une caméra linéaire de 64 pixels sur 256 niveaux de gris. Nous utiliserons ce capteur de manière très spécifique comme détecteur d'incendie dans la tâche de surveillance présentée au Chapitre 10.

Récapitulatif des différentes variables de bases disponibles :

Nom	Désignation	Domaine	Pas
`Vg`	Vitesse roue gauche	{-20, …, +20}	1
`Vd`	Vitesse roue droite	{-20, …, +20}	1
`Vrot`	Vitesse rotation	{-10, …, +10}	1
`Vtrans`	Vitesse translation	{-10, …, +10}	1

Tableau 3.3 — Variables motrices de bases.

Nom	Désignation	Domaine	Pas
`Px0` à `Px7`	Capteurs de proximité	{0, …1023}	1
`Lm0` à `Lm7`	Capteurs de luminosité	{0, …511}	1
`PosG`	Position roue gauche	{-32000, …32000}	1
`PosD`	Position roue droite	{-32000, …32000}	1

Tableau 3.4 — Variables sensorielles de bases.

3.2 L'environnement

L'environnement habituel du robot Khepera est une plate-forme de 1m x 1m. Les parois de l'environnement sont texturées pour être facilement visibles par la caméra linéaire. La texture est donnée par les lettres qui composent la citation de John W. Tukey : *« Far better an approximate answer to the right question which is often vague, than an exact answer to the wrong question which can always be made precise* » (voir Figure 3-4).

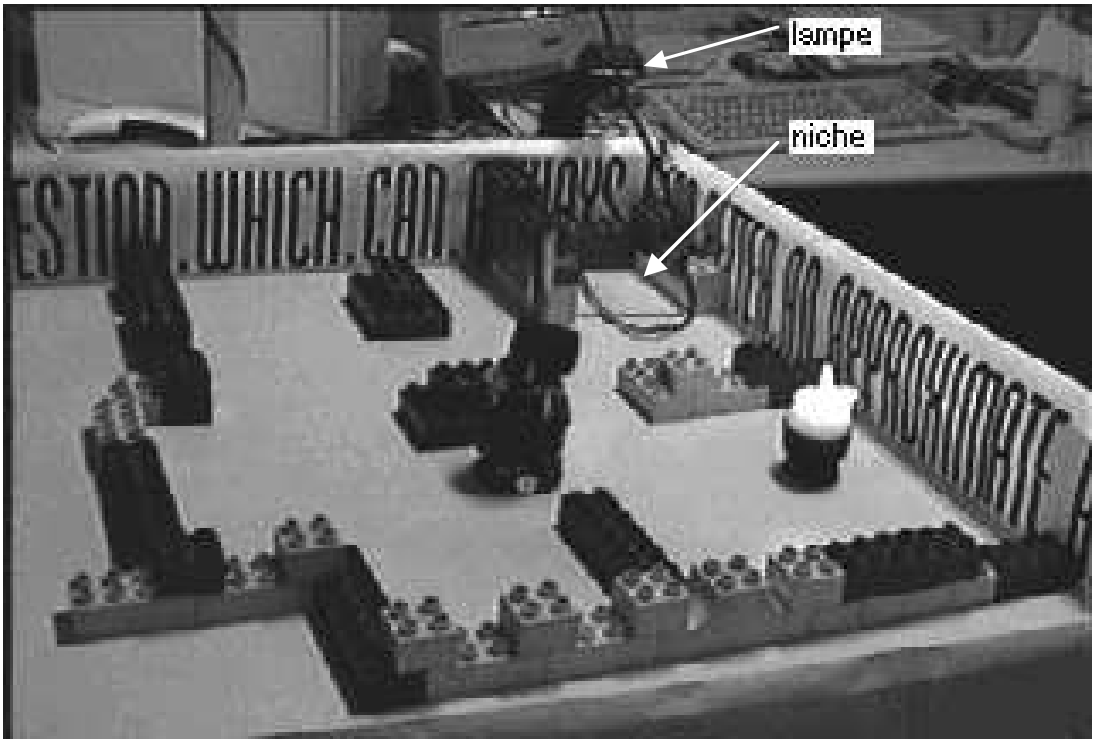


Figure 3-4 — L'environnement du Khepera

Les obstacles, les murs et les couloirs sont généralement constitués de gros blocs de Lego© de couleurs et de dimensions variées. La présence d'objets de nature différente (tasse à café, balle de tennis, petits jouets divers, etc.) est cependant possible et, en général, ne perturbe pas le fonctionnement correct du robot.

L'éclairage habituel de l'environnement est celui d'un bureau standard : une fenêtre et des tubes néons au plafond. Le robot peut s'adapter à des variations considérables de cet éclairement. Ses capteurs infrarouges de proximités sont cependant fortement perturbés par un éclairage très intense (notamment solaire). Le robot est alors complètement « ébloui » et n'arrive plus à « voir » correctement murs et obstacles. Inversement, une luminosité très basse ne perturbe en rien les capteurs de proximité infrarouges, mais peut bien évidemment gêner la caméra linaire dont la vitesse d'échantillonnage peut devenir très faible.

Pour certaines expériences, il a été construit dans l'environnement une « niche » pour le robot. Cette niche est située dans un coin et délimitée par des parois en Lego©. Cette niche est surmontée d'une petite lampe qui induit dans l'environnement un faible gradient de lumière qui aide le robot à retrouver sa base (voir Figure 3-4 en haut à droite).

3.3 Contrôle des flux sensoriels et moteurs

Le module de contrôle a pour fonction la gestion du flux des données sensorielles, une éventuelle mémorisation, l'appel aux différents calculs (mise à jour des pré-traitements sur les données brutes, calcul des comportements réflexes) et enfin l'envoi des ordres moteurs. Le flux de données est imposé par le robot via une communication asynchrone. Avec ce protocole, le Khepera envoie les données sensorielles vers l'ordinateur périodiquement et automatiquement. Ce flux est contrôlé par l'ordinateur par l'intermédiaire d'ordres spécifiques envoyés au Khepera indiquant le type des données, la périodicité et le nombre d'envois désirés. Nous travaillons avec une période d'échantillonnage de 100 ms (10 Hz).

Le module de contrôle se synchronise sur l'arrivée de nouvelle donnée. Nous avons défini un squelette type pour le module contrôle présenté Figure 3-5.

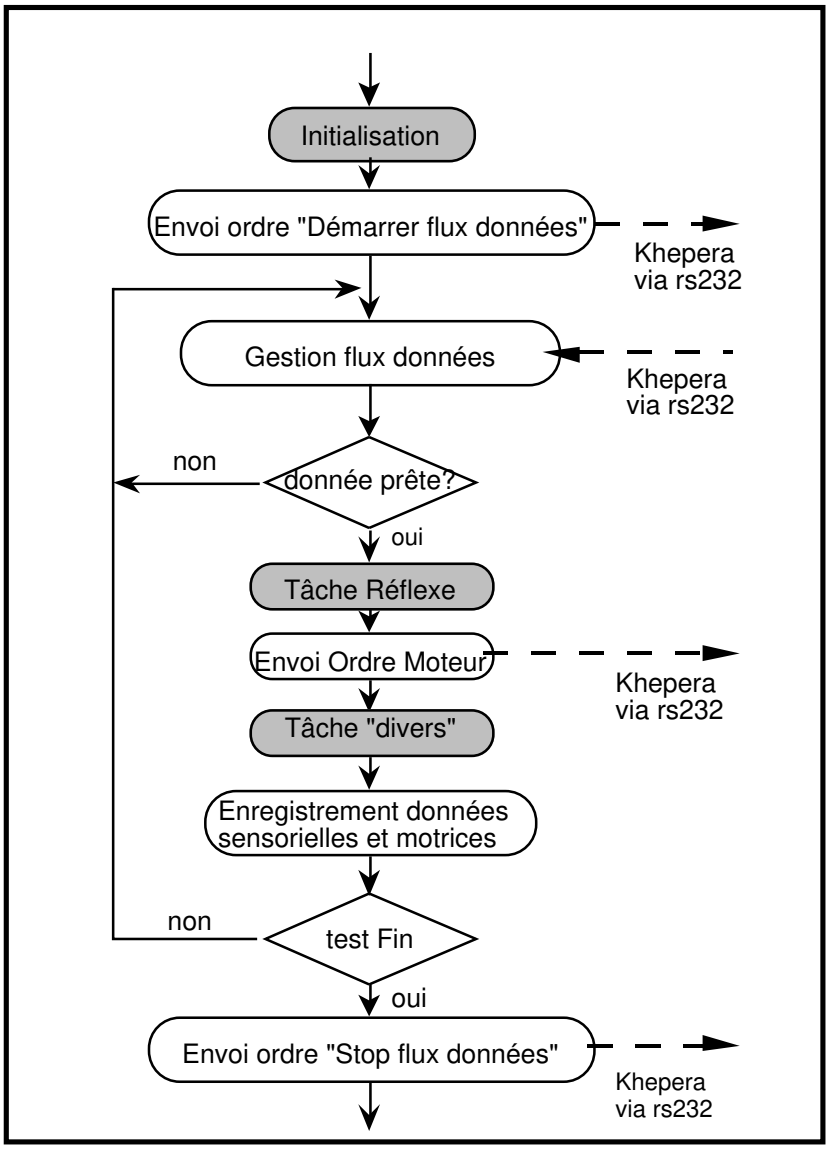


Figure 3-5 — Schéma du squelette type d'un programme de contrôle du robot.

Les parties grisées correspondent aux modules devant être spécifiés par le programmeur. Ils sont au nombre de trois. Le premier est le module d'initialisation. Le second est le module « Tâche réflexe ». Il est exécuté dès la réception de nouvelles données. Dans ce module, le

programmeur définit les calculs nécessaires pour la mise à jour des commandes motrices en prenant en compte les valeurs des données qui viennent d'arriver.

Le dernier module « Tâches divers » permet au programmeur de spécifier différentes tâches comme la mise à jour de modèles paramétriques lors de tâches d'apprentissage, le calcul de différentes inférences probabiliste, la gestion d'une interface graphique…

Ce squelette est très simple. Les différentes expériences présentées dans ce document ont été réalisées sur la base de ce squelette.

3.4 Exécution d'un programme probabiliste.

Pour un programme donné, l'interpréteur est substitué à la partie tâche réflexe. Cet interpréteur calcule le vecteur correspondant aux variables motrices.

Prenons l'exemple du programme probabiliste réflexe du chapitre précédent : les ordres moteurs `M` sont déterminés par la situation sensorielle courante `S` selon la description *Cp_tâche_reflexe :*

$$P(\mathtt{M}\ \mathtt{S} \mid \textit{Cp_tâche_reflexe}) = P(\mathtt{M} \mid \mathtt{S}\ \textit{Cp_tâche_reflexe})\ \ P(\mathtt{S} \mid \textit{Cp_tâche_reflexe}).$$

Le programme repose sur l'utilisation de cette description en posant la question probabiliste suivante :

$$P(\mathtt{M} \mid \mathtt{S}\ \textit{Cp_tâche_reflexe}).$$

La décision pour fixer la valeur de la variable motrice sera généralement obtenue par un tirage aléatoire selon la distribution $P(\mathtt{M} \mid \mathtt{S}\ \textit{Cp_tâche_reflexe})$.

En pratique, l'interpréteur utilise une fonction spécifique (Prob_`M`|`S`_*Cp_tâche_reflexe* ($\mathtt{m}_i\ \mathtt{s}_j$)) qui retourne la probabilité que la variable $\mathtt{M} = \mathtt{m}_i$ sachant que la variable $\mathtt{S} = \mathtt{s}_j$. La mise à jour des variables sensorielles `S` est faite périodiquement. À chaque variable sur laquelle repose la question (par exemple ici `M`) est associée une table de valeurs réelles (valp_M[k_M]) représentant la distribution résultant de la question. La mise à jour de cette table est réalisée périodiquement en utilisant cette fonction. Le tirage aléatoire suivant la distribution obtenue est implémentée très classiquement. Il consiste à parcourir la table des probabilités (valp_M[k_M]) tant qu'une valeur aléatoire tirée entre 0.0 et 1.0 est inférieure au cumul des probabilités parcourues.

L'exemple suivant illustre le code définissant le module réflexe du squelette de contrôle présenté section 3.3 :

```
 (defun Reflexe

;;; maj de la variable sensorielle, calcul des pré-traitements
;;; à partir des données sensorielle brutes.
 S ← sj ;

;;; maj de la distribution de probabilité sur la variable M :
;;; P(M | S Cp_tâche_reflexe)
 Pour chaque valeur mi de M
   valp_M[i] ← Prob_M|S_Cp_tâche_reflexe(mi, sj)
 fin Pour

;;; Décision de la valeur de M par tirage probabiliste
;;; tirer une valeur entre 0.0 et 1.0
 alea ← random(1.0)
 i ← 0
 cumul_prob ← valp_M[0]

 Tantque (cumul_prob < alea)
    i ← i + 1
    cumul_prob ← cumul_prob + valp_M[i]
 fin Tantque

 M ← mi

; envoi de l'ordre vers le Khepera
 Envoyer-Ordre(M) ;

)
```

3.5 Télé-opération

Il est possible de télé-opérer le robot Khepera à partir d'un manche à balai de jeu vidéo. Comme nous le verrons cette télé-opération joue un rôle très important dans certains apprentissages. Le procédé de télé-opération est le suivant. Le robot peut fonctionner suivant trois modes :

- Le mode TEL : « télé-opéré sans apprentissage » ;
- Le mode APP : « télé-opéré avec apprentissage » ;
- Le mode RES : « restitution (par inférence) du comportement appris ».

Dans le mode TEL le robot se déplace en fonction des commandes motrices envoyées par l'opérateur via le manche à balai. Ce mode permet notamment de déplacer le robot dans l'environnement et de le mettre dans des situations sensori-motrices intéressantes.

Dans le mode APP comme dans le mode TEL, les commandes motrices, données via le manche à balai, sont appliquées directement aux moteurs. Dans un même temps, pour le mode APP, les données sensorielles (échantillonnées à 10 Hz) sont associées à ces ordres moteurs et permettent la constitution de jeux de données sensori-motrices en vue de la mise à jour des paramètres du modèle.

En mode RES, les commandes motrices M sont obtenues par un tirage aléatoire selon la distribution conditionnée par les valeurs des variables sensorielles S : P(M | S *D C*).

L'opérateur n'intervient plus. Le robot décide seul de ce qu'il doit faire en fonction des apprentissages probabilistes qu'il a fait précédemment.

Les boutons du manche à balai permettent de passer d'un mode à l'autre à volonté. Ainsi l'apprentissage peut être à tout moment interrompu et testé (mode RES). Nous utilisons le mode télé-opéré (TEL) afin de placer le robot dans des situations particulières en inhibant l'apprentissage ; une fois positionné, nous pouvons alors basculer en mode apprentissage afin d'indiquer au Khepera que faire dans de telles situations.

Chapitre 4

Comportements réactifs

Nous présentons ici différents comportements réactifs et la manière dont nous les avons réalisés sous la forme de *descriptions*. L'objectif poursuivi est de fournir au robot Khepera un ensemble de comportements simples lui permettant d'interagir avec son environnement.

4.1 Phototaxie

4.1.1 Objectifs et protocole expérimental

Pour cette première expérience, nous désirons fournir au Khepera la capacité de se diriger vers une source lumineuse (comportement dit de phototaxie) sous la forme d'un comportement réactif. Nous allons construire une description très simple mettant en relation une seule variable motrice (`Vrot`) et une seule variable sensorielle (`Lum`). Nous testerons ensuite cette description en situation réelle.

4.1.2 Spécification

4.1.2.1 Choix des variables

Les variables motrices à notre disposition sont `Vrot` et `Vtrans`. Pour ce comportement, nous avons décidé de ne travailler qu'avec la vitesse de rotation ; nous supposons ainsi que le fait de se diriger vers la lumière revient à choisir une vitesse de rotation.

Le Khepera dispose de huit capteurs de luminosité notés `Lm0` à `Lm7`. Chacun de ces capteurs retourne une valeur comprise entre 0 (luminosité forte) et 511 (luminosité faible). Le nombre de situations sensorielles discernables est très grand (512^8 cas possibles). Nous avons

choisi de résumer cette information sous la forme d'une seule variable : Lum. Cette variable sensorielle est un pré-traitement appliqué sur les variables sensorielles brutes Lmi, elle retourne la valeur angulaire correspondant à l'indice de la cellule la plus éclairée. Elle peut prendre ainsi 8 valeurs distinctes : {-170, -90, -45, -10, +10, +45, +90, +170}.

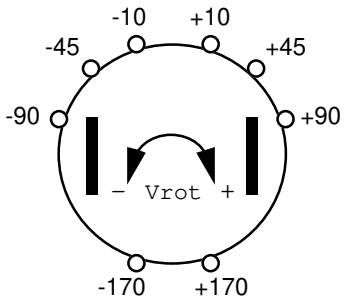


Figure 4-1 — Variables pertinentes pour le comportement phototaxie.
Vrot (-10 à 10) et Lum {-170, -90, -45, -10, +10, +45, +90, +170}.

Variables pertinentes

- **Lum** : domaine {-170, -90, -45, -10, +10, +45, +90, +170}, $k_{\mathtt{Lum}} = 8$;
- **Vrot** : domaine {-10, -9,…,+10}, $k_{\mathtt{Vrot}} = 21$.

4.1.2.2 Connaissances préalables de dépendance

La description du comportement porte sur les variables Lum et Vrot et correspond à la distribution conjointe :

$$\mathrm{P}(\mathtt{Lum}\ \mathtt{Vrot} \mid D\ C), \tag{4.1}$$

que nous décomposons sous la forme du produit de deux distributions élémentaires :

$$\begin{aligned} &\mathrm{P}(\mathtt{Lum}\ \mathtt{Vrot} \mid D\ C) \\ &= \mathrm{P}(\mathtt{Lum} \mid D\ C)\mathrm{P}(\mathtt{Vrot} \mid \mathtt{Lum}\ D\ C). \end{aligned} \tag{4.2}$$

4.1.2.3 Connaissances préalables d'observation

Nous choisissons de ne pas privilégier certaines situations sensorielles vis-à-vis d'autres (nous considérons que la source lumineuse peut être dans n'importe quelle direction par rapport au robot) et nous ne souhaitons pas observer leur répartition expérimentale :

$$\mathrm{P}\big([\mathtt{Lum} = \mathtt{lum}_i] \mid D\ C\big) = \mathrm{P}\big([\mathtt{Lum} = \mathtt{lum}_i] \mid C\big) = \mathbf{U}_{k_{\mathtt{Lum}}} = \frac{1}{8}. \tag{4.3}$$

Nous choisissons de représenter la probabilité de Vrot conditionnée par Lum sous la forme d'un ensemble de lois normales paramétrées par la valeur de Lum :

$$P(\texttt{Vrot} \mid [\texttt{Lum} = \texttt{lum}_i] D C) = \mathbf{G}_{\mu[\texttt{lumi}],\, \sigma[\texttt{lumi}]}(\texttt{Vrot}). \quad (4.4)$$

4.1.3 Identification

Pour cette expérience, nous avons fixé les valeurs des paramètres « à la main », c'est-à-dire sans aucune donnée expérimentale. Intuitivement, les espérances et écart-types doivent êtres choisis pour mener au comportement suivant :

- Plus l'orientation du robot est écartée de la source lumineuse plus sa vitesse de rotation doit être rapide et moins les contraintes sur la valeur exacte des vitesses sont importantes (c'est-à-dire plus l'écart-type est important). La vitesse est positive si la source est à sa gauche, négative sinon.
- Lorsque que le robot se trouve face à la source, la variable `Lum` peut valoir soit -10, soit +10 (le Khepera ne disposant pas de cellule située exactement de face). Pour ces deux situations, nous imposons au robot une faible rotation (négative pour `Lum` = +10 et positive pour `Lum` = -10) de façon à maintenir globalement la source entre ces deux capteurs. La notion intuitive « d'imposer » se traduit par une plus forte contrainte sur les vitesses (c'est-à-dire un écart type plus faible).
- Lorsque le robot tourne le dos à la source lumineuse, les cellules les plus activées sont soit la cellule n°7 (`Lum` = -170), soit la cellule n°6 (`Lum` = +170). Dans ce cas, nous voulons tourner du côté le plus éclairé (positivement pour `Lum` = +170 et négativement pour `Lum` = -170).

Le Tableau 4.1 ci-dessous présente les espérances μ[`lumi`] et écart-types σ[`lumi`] fixés pour chacune des valeurs de `Lum` et la Figure 4-2 en donne une représentation graphique.

`lumi`	μ[`lumi`]	σ[`lumi`]
-170	-8.5	2.0
-90	-6.5	1.5
-45	-4.0	1.0
-10	-1.0	0.5
+10	+1.0	0.5
+45	+4.0	1.0
+90	+6.5	1.5
+170	+8.5	2.0

Tableau 4.1 — Espérances et écart-types pour P(`Vrot` | `Lum`).

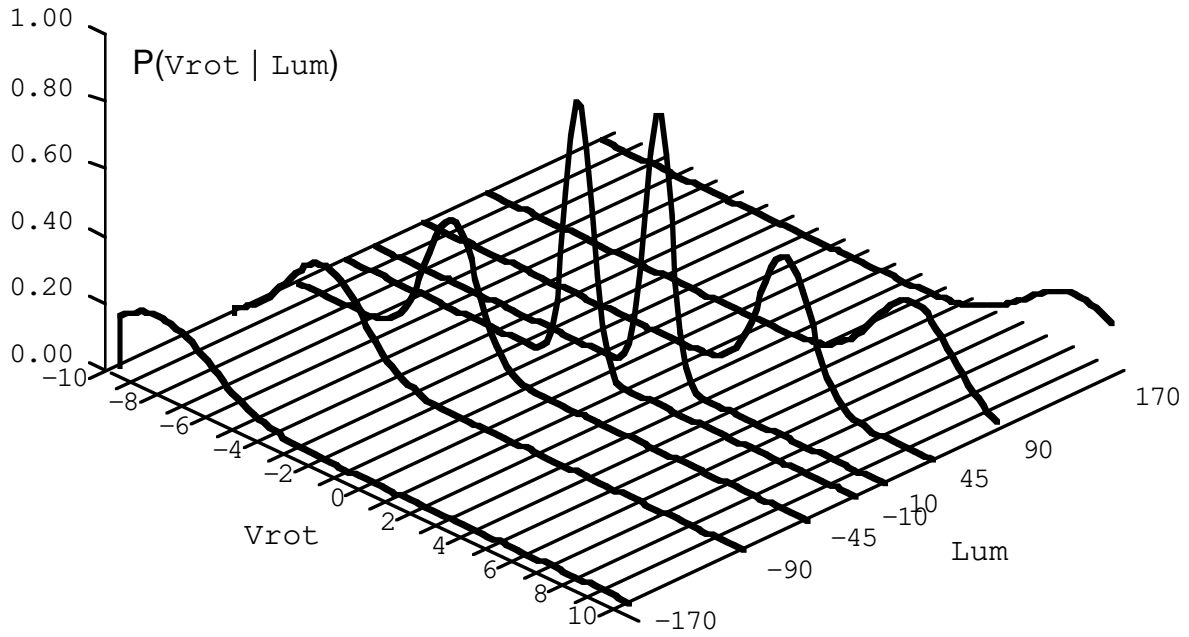


Figure 4-2 — Visualisation de P(Vrot | Lum *Cp_Photo*) pour les différentes valeurs de Lum.

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_Photo* :

$$\begin{aligned} &\mathrm{P}(\mathtt{Lum}\ \mathtt{Vrot} \mid Cp_Photo) \\ &= \mathrm{P}(\mathtt{Lum} \mid Cp_Photo)\mathrm{P}(\mathtt{Vrot} \mid \mathtt{Lum}\ Cp_Photo). \end{aligned} \tag{4.5}$$

4.1.4 Utilisation

4.1.4.1 Programme et question probabiliste

Nous rappelons qu'utiliser cette description pour faire fonctionner le robot Khepera en mode « phototaxie » va consister à :

- Mettre à jour la valeur de la variable Lum à partir des valeurs des capteurs de lumières (fréquence 10 Hz).
- Tirer au hasard une valeur de Vrot suivant la distribution P(Vrot | Lum *Cp_Photo*) de la distribution correspondant à la connaissance préalable *Cp-Photo*.
- Envoyer la valeur de Vrot ainsi trouvée en consigne aux moteurs du robot.

Le programme de commande est décrit par :

1/ Description utilisée :

Cp_Photo

2/ Requête :

E_q : {Vrot}

E_c : {Lum}

E_i : Ø

Question : P(Vrot | Lum *Cp_Photo*)

3/ Décision :

Tirage aléatoire

4.1.4.2 Résultat

Le comportement phototaxie obtenu est très satisfaisant. Le robot se tourne très rapidement vers la source lumineuse indépendamment de son orientation initiale et de l'intensité lumineuse.

Nous avons testé ce modèle pour différentes vitesses positives de translation sans constater une dégradation du comportement.

Nous avons observé un phénomène attendu d'oscillation face à la lumière (la valeur de Lum oscille entre -10 et +10) permettant de maintenir globalement le cap vers la source. Toutefois l'oscillation s'est révélée beaucoup moins prononcée que prévue.

Nous avons également testé ce comportement en situation d'éclairage diffus (les néons du plafond) plutôt que ponctuel. Ici encore le comportement du robot est stable, il se déplace aléatoirement en évitant les zones d'ombre.

4.1.5 Enseignements et discussion

La première chose à retenir de cette expérience est la simplicité d'utilisation des descriptions. L'utilisation de la description de phototaxie se résume à poser la même question, tous les dixièmes de secondes, au moteur d'inférence.

La programmation de ce comportement illustre bien l'importance pour le concepteur de travailler sur un pré-traitement des données sensorielles qui soit facilement interprétable. Le fait de jouer dans le modèle sur les incertitudes, pour ce cas, nous dispense d'une modélisation analytique détaillée reposant sur un calibrage fin des capteurs. Un pré–traitement "intuitif" et empirique suffit. Ce point est important ; nous y reviendrons notamment en présentant l'évitement d'obstacle.

Nous retiendrons également l'intérêt de traduire le niveau de contrainte sur les vitesses par un écart-type. Généralement l'incertitude d'un modèle est associée à la notion de "bruit". Ici, nous utilisons l'incertitude (les écart-types) plutôt comme la quantification du niveau de contrainte sur les ordres moteurs selon la situation sensorielle. Cette idée sera à nouveau illustrée avec l'expérience sur l'évitement d'obstacle (section 4.3.1).

4.2 Apprentissage de comportements réactifs

4.2.1 Objectifs et protocole expérimental

L'objectif de cette deuxième expérience est de construire un ensemble de comportements réactifs (éviter les obstacles, pousser les obstacles et suivre les contours d'un obstacle) obtenus sur la base d'une dépendance entre les informations de proximité et la vitesse de rotation du Khepera. Nous montrons que, sur la base d'un unique ensemble de connaissances préalables, il est possible d'identifier et de restituer différents comportements réactifs selon le jeu de données. On peut ainsi obtenir différents comportements éventuellement antagonistes.

La démarche que nous suivons est la suivante. Dans un premier temps, nous spécifions les connaissances préalables qui nous semblent pertinentes pour identifier ces différents comportements réactifs associés aux informations fournies par les capteurs de proximité. Puis nous allons soumettre ce même ensemble de connaissances préalables à différents ensembles de données expérimentales. Chacun de ces ensembles de données est obtenu par l'enregistrement des valeurs des variables motrices et sensorielles au cours d'une télé-opération. La télé-opération consiste à piloter le Khepera de visu en contrôlant les ordres moteurs à l'aide d'un manche à balai. Au cours de la télé–opération, l'opérateur contrôle le robot de façon à lui faire exécuter un des trois comportements.

4.2.2 Spécification

4.2.2.1 Choix des variables

Les variables motrices à notre disposition sont la vitesse de translation (`Vtrans`) et la vitesse de rotation (`Vrot`). Nous avons décidé que notre description ne porterait que sur la vitesse de rotation. Ce choix peut paraître contestable. En effet, il semble naturel de penser que, dans le cadre d'un comportement d'évitement d'obstacle, devant un obstacle proche il soit prudent de ralentir. Toutefois, pour des raisons de simplicité et de facilité de contrôle par télé-opération, le comportement décrit ici ne prend pas en compte cet aspect. Ainsi l'identification de ce comportement se fera avec une vitesse de translation faible (fixé à +3, correspondant à une vitesse de translation du robot de 24mm.s^{-1}). Nous indiquerons un peu plus loin comment moduler la vitesse de translation en fonction de la proximité des obstacles (voir la section 4.3.2).

Le Khepera dispose de huit capteurs de proximités, notés `Px0`…`Px7`. Chacun d'eux retourne une valeur comprise entre 0 et 1023. Le nombre de situations sensorielles discernables est très grand (1024^8 cas possibles). Nous avons choisi de résumer cette information sous la forme de deux variables : `Dir` et `Prox`. Ces deux variables sensorielles sont le résultat de pré-traitements appliqués aux six capteurs de proximité avant (`Px0` à `Px5`). Nous pouvons les interpréter respectivement comme la direction et la proximité de l'obstacle le plus proche du robot.

$$\texttt{Dir} = \text{Floor}\left(\frac{90(\texttt{Px5}-\texttt{Px0})+45(\texttt{Px4}-\texttt{Px1})+5(\texttt{Px3}-\texttt{Px2})}{9\left(1+\sum_{i=0}^{5}\texttt{Pxi}\right)}\right) \tag{4.6}$$

$$\texttt{Prox} = \text{Floor}\left(\frac{\max(\texttt{Px0},\ \texttt{Px1},\ \texttt{Px2},\ \texttt{Px3},\ \texttt{Px4},\ \texttt{Px5})}{64}\right) \tag{4.7}$$

La variable `Dir` prend des valeurs entières entre -10 (obstacle à gauche du robot) et +10 (obstacle à la droite du robot) soit 21 cas possibles ; la variable `Prox` prend des valeurs entières entre 0 (pas d'obstacle) et 15 (obstacle très proche du robot). La Figure 4-3 illustre une situation pour laquelle `Dir` = -4 et `Prox` = 3.

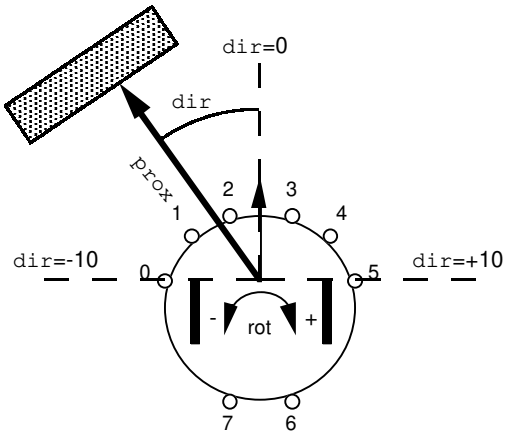


Figure 4-3 — Pré-traitements sur les capteurs de proximité avant, ici `Dir` = -4 et `Prox` = 3.

Variables pertinentes

- **`Dir`** : domaine {-10, -9,…, +10}, $k_{\texttt{Dir}} = 21$;
- **`Prox`** : domaine {0, 1,…, 15}, $k_{\texttt{Prox}} = 16$;
- **`Vrot`** : domaine {-10, -9,…, +10}, $k_{\texttt{Vrot}} = 21$.

4.2.2.2 Connaissances préalables de dépendance

La description du comportement porte sur les variables `Dir`, `Prox` et `Vrot` et correspond à la distribution conjointe :

$$P(\texttt{Prox}\ \texttt{Dir}\ \texttt{Vrot} \mid D\ C). \tag{4.8}$$

Nous allons ici exprimer des connaissances nous permettant de décomposer cette distribution conjointe en un produit de distributions élémentaires. En appliquant la règle **[R1]** nous pouvons décomposer la distribution conjointe selon les treize manières suivantes :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Prox Dir Vrot} \mid D\ C)\\
&= \mathrm{P}(\texttt{Prox} \mid D\ C)\mathrm{P}(\texttt{Dir Vrot} \mid \texttt{Prox}\ D\ C)\\
&= \mathrm{P}(\texttt{Dir} \mid D\ C)\mathrm{P}(\texttt{Prox Vrot} \mid \texttt{Dir}\ D\ C)\\
&= \mathrm{P}(\texttt{Vrot} \mid D\ C)\mathrm{P}(\texttt{Prox Dir} \mid \texttt{Vrot}\ D\ C)\\
&= \mathrm{P}(\texttt{Prox Dir} \mid D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ D\ C)\\
&= \mathrm{P}(\texttt{Prox} \mid D\ C)\mathrm{P}(\texttt{Dir} \mid \texttt{Prox}\ D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ D\ C)\\
&= \mathrm{P}(\texttt{Dir} \mid D\ C)\mathrm{P}(\texttt{Prox} \mid \texttt{Dir}\ D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ D\ C)\\
&= \mathrm{P}(\texttt{Prox Vrot} \mid D\ C)\mathrm{P}(\texttt{Dir} \mid \texttt{Prox Vrot}\ D\ C)\\
&= \mathrm{P}(\texttt{Prox} \mid D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox}\ D\ C)\mathrm{P}(\texttt{Dir} \mid \texttt{Prox Vrot}\ D\ C)\\
&= \mathrm{P}(\texttt{Vrot} \mid D\ C)\mathrm{P}(\texttt{Prox} \mid \texttt{Vrot}\ D\ C)\mathrm{P}(\texttt{Dir} \mid \texttt{Prox Vrot}\ D\ C)\\
&= \mathrm{P}(\texttt{Dir Vrot} \mid D\ C)\mathrm{P}(\texttt{Prox} \mid \texttt{Dir Vrot}\ D\ C)\\
&= \mathrm{P}(\texttt{Dir} \mid D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Dir}\ D\ C)\mathrm{P}(\texttt{Prox} \mid \texttt{Dir Vrot}\ D\ C)\\
&= \mathrm{P}(\texttt{Vrot} \mid D\ C)\mathrm{P}(\texttt{Dir} \mid \texttt{Vrot}\ D\ C)\mathrm{P}(\texttt{Prox} \mid \texttt{Dir Vrot}\ D\ C).
\end{aligned}
\tag{4.9}
$$

Parmi les treize expressions de la distribution conjointe, nous décidons de décomposer (4.8) de la manière suivante :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Prox Dir Vrot} \mid D\ C)\\
&= \mathrm{P}(\texttt{Prox Dir} \mid D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ D\ C).
\end{aligned}
\tag{4.10}
$$

Nous pensons qu'il n'existe pas a priori de dépendance entre la direction dans laquelle on observe un obstacle et sa proximité. Nous traduisons cette hypothèse en spécifiant que les variables `Prox` et `Dir` sont indépendantes (conditionnellement aux connaissances C) :

$$
\mathrm{P}(\texttt{Prox Dir} \mid D\ C) = \mathrm{P}(\texttt{Prox} \mid D\ C)\mathrm{P}(\texttt{Dir} \mid D\ C). \tag{4.11}
$$

Nous obtenons finalement la décomposition suivante :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Prox Dir Vrot} \mid D\ C)\\
&= \mathrm{P}(\texttt{Prox} \mid D\ C)\mathrm{P}(\texttt{Dir} \mid D\ C)\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ D\ C).
\end{aligned}
\tag{4.12}
$$

4.2.2.3 Connaissances préalables d'observation

Nous venons de décomposer la distribution conjointe en un produit de distributions élémentaires. À ce stade de spécification des connaissances préalables il reste à indiquer les valeurs des distributions élémentaires et la manière dont elles évoluent au vue de données expérimentales.

Nous considérons que les obstacles peuvent être rencontrés suivant n'importe quelle direction et à n'importe quelle distance. De plus, dans le cadre de cette expérience, nous ne chercherons pas à faire évoluer leurs probabilités selon les données expérimentales. Nous traduisons ces choix par :

$$\begin{aligned} &P\left(\left[\texttt{Prox} = \texttt{prox}_i\right] \mid D\ C\right) = P\left(\left[\texttt{Prox} = \texttt{prox}_i\right] \mid C\right) = \mathbf{U}_{k\texttt{Prox}} = \frac{1}{16} \\ &P\left(\left[\texttt{Dir} = \texttt{dir}_j\right] \mid D\ C\right) = P\left(\left[\texttt{Dir} = \texttt{dir}_j\right] \mid C\right) = \mathbf{U}_{k\texttt{Dir}} = \frac{1}{21}. \end{aligned} \tag{4.13}$$

Enfin, nous décidons de représenter la probabilité de `Vrot` conditionnée par `Dir` et `Prox` sous la forme d'une famille de lois de Gauss paramétrées par les valeurs de `Dir` et `Prox` :

$$\begin{aligned} &P\left(\texttt{Vrot} \mid \left[\texttt{Prox} = \texttt{prox}_i\right]\left[\texttt{Dir} = \texttt{dir}_j\right] D\ C\right) \\ &= \mathbf{G}_{\mu[\texttt{proxi},\texttt{dirj}],\ \sigma[\texttt{proxi},\texttt{dirj}]}(\texttt{Vrot}). \end{aligned} \tag{4.14}$$

4.2.3 Identification

Les trois expériences que nous présentons reposent sur l'unique connaissance préalable énoncée ci-dessus. En exploitant ce modèle, nous avons appris trois comportements distincts en le confrontant simplement à trois jeux de données différents, obtenus en télé-opérant successivement le robot pour éviter les obstacles, pour pousser les obstacles et pour suivre les contours.

Le mécanisme d'identification adopté est un apprentissage « par cœur », sans généralisation entre valeurs voisines de `Dir` et `Prox` : pour chaque situation sensorielle [`Dir`, `Prox`] nous prenons pour μ[`Dir`, `Prox`] et σ[`Dir`, `Prox`] les moyennes et écart-types expérimentaux des commandes motrices `Vrot`. Pour déterminer ces valeurs itérativement, nous procédons ainsi : pour chaque valeur de [`Dir`, `Prox`] nous mémorisons :

- n : le nombre de fois où le robot s'est retrouvé dans cette situation lors de l'apprentissage ;
- S_1 : la somme des valeurs de `Vrot` obtenues dans cette situation : $S_1 = \sum$`Vrot` ;
- S_2 : la somme des carrés de ces valeurs : $S_2 = \sum$`Vrot`^{2}.

Ces valeurs sont facilement mise à jour à chaque nouvelle mesure, permettant ainsi une forme d'apprentissage incrémental. Les paramètres μ[`Dir`, `Prox`] et σ[`Dir`, `Prox`] sont déduits par :

$$\begin{aligned} \mu[\texttt{Dir}, \texttt{Prox}] &= \frac{S_1[\texttt{Dir}, \texttt{Prox}]}{n[\texttt{Dir}, \texttt{Prox}]} \\ \sigma[\texttt{Dir}, \texttt{Prox}] &= \sqrt{\frac{S_2}{n[\texttt{Dir}, \texttt{Prox}]} - \mu^2[\texttt{Dir}, \texttt{Prox}]}\ . \end{aligned} \tag{4.15}$$

Nous initialisons les valeurs de n, S_1, S_2 respectivement par 0,01, 0, 0,36, fixant ainsi les valeurs des paramètres en absence de données à $\mu = 0$ et $\sigma = 6$; correspondant à un a priori proche de l'uniforme, rapidement « oublié » (n=0,01) par la prise en compte des données expérimentales.

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_Obstacle_Cpt_x*, qui correspond à l'appariement de la connaissance préalable *Cp_Obstacle_Cpt* et d'un jeux de données expérimentales particulier *D_x* obtenu en télé-opérant le Khepera en lui faisant suivre un comportement x :

$$
\begin{aligned}
&P(\texttt{Prox Dir Vrot} \mid Cp_Obstacle_Cpt_x) \\
&P(\texttt{Prox Dir Vrot} \mid D_x\ Cp_Obstacle) \\
&= \left| \begin{array}{l} P(\texttt{Prox} \mid Cp_Obstacle) \\ P(\texttt{Dir} \mid Cp_Obstacle) \\ P(\texttt{Vrot} \mid \texttt{Prox Dir}\ D_x\ Cp_Obstacle). \end{array} \right.
\end{aligned} \tag{4.16}
$$

4.2.4 Utilisation

4.2.4.1 Programme et question probabiliste

Nous avons appliqué cette méthode pour apprendre par télé-opération trois comportements réflexes : « éviter les obstacles », « pousser les obstacles » et « suivre le contour d'un obstacle ».

Comme il est indiqué au chapitre 3 section 3.4, le robot peut fonctionner suivant trois modes :

- Le mode TEL : « télé-opéré sans apprentissage » ;
- Le mode APP : « télé-opéré avec apprentissage » ;
- Le mode RES : « restitution (par inférence) du comportement appris ».
- Pour chaque comportement x appris, nous associons un jeu de paramètres particulier, μ_x[Dir, Prox] et σ_x[Dir, Prox] qui est mis à jour au cours de l'apprentissage.

Le programme d'apprentissage et de restitution pour un comportement donné va consister à :

- Mettre à jour la valeur des variables sensorielles Dir et Prox à partir des valeurs des capteurs de proximité (fréquence 10 Hz).
- Selon le mode de contrôle du robot :
  - Mode TEL : « télé-opéré sans apprentissage »,

    Lire la position du manche à balai et en déduire une consigne pour Vrot. Envoyer la valeur de Vrot ainsi obtenue en consigne aux moteurs du robot.
  - Mode APP : « télé-opéré avec apprentissage »,

    Lire la position du manche à balai et en déduire une consigne pour Vrot. Envoyer la valeur de Vrot ainsi obtenue en consigne aux moteurs du robot. Mettre à jour les paramètres μ_x[Dir Prox] et σ_x[Dir, Prox] de la distribution P(Vrot | Dir Prox *Cp_Obstacle_Cpt_x*).
  - Mode RES « restitution (par inférence) du comportement appris »

    Tirer au hasard une valeur de Vrot suivant la distribution P(Vrot | Dir Prox) de l'une des trois descriptions correspondant à l'appariement de la connaissance préalable *Cp_Obstacle* et de l'un des trois jeux de données *D_éviter*, *D_pousser* ou *D_suivre*. Envoyer la valeur de Vrot ainsi obtenue en consigne aux moteurs du robot.

Le programme de restitution pour faire fonctionner le robot Khepera selon l'un des comportements des trois expériences se résume à sélectionner une des trois descriptions correspondant à l'appariement de la connaissance préalable `Cp_Obstacle` et de l'un des trois jeux de données *D_éviter*, *D_pousser* ou *D_suivre* :

1/ Description utilisée : selon le type de comportement télé-opéré :

soit *Cp_Obstacle_Cpt_ éviter*

soit *Cp_Obstacle_Cpt_pousser*

soit *Cp_Obstacle_Cpt_suivre*

2/ Requête :

E_q : {Vrot}

E_c : {Dir, Prox}

E_i : Ø

Question : P(Vrot | Dir Prox *Cp_Obstacle_Cpt_x*)

3/ Décision :

Tirage aléatoire

4.2.4.2 Résultat

Éviter les obstacles

L'apprentissage de l'évitement d'obstacles consiste à télé-opérer de visu le robot dans un environnement encombré, en évitant le contact avec les obstacles. Quelques secondes d'apprentissage (mode APP) permettent de lui apprendre à aller tout droit en espace libre. Les phases d'apprentissage et de restitution (mode RES) se succèdent, permettant ainsi de juger le comportement en cours d'expérimentation. Très rapidement le robot a déjà un comportement reconnaissable. Après une à deux minutes (correspondant à environ 1200 données), nous constatons que la restitution du comportement devient très correcte. Le robot est capable d'éviter des obstacles se présentant latéralement ou de face, indépendamment de leurs formes, de leurs orientations, et, dans une certaine mesure, de leur mouvement.

L'analyse de la loi de probabilité sur `Vrot` ainsi identifiée montre que pour des valeurs de `Dir` négatives (obstacle à gauche), la valeur du paramètre μ est d'autant plus positive (rotation sur la droite) que `Prox` est élevée (obstacle proche) et symétriquement pour les valeurs de `Dir` positives.

Lorsque l'obstacle se trouve face au robot (`Dir` = 0), nous avons remarqué que le paramètre μ prenait des valeurs proches de 0 (rotation nulle !) et était associé à des valeurs de σ importantes (traduisant une forte incertitude). Ce phénomène s'explique très bien. En effet, en présence d'un obstacle face au robot, les réponses de l'opérateur montrent deux pics : tourner sur la droite ou sur la gauche (Figure 4-4). Le modèle gaussien conduit à moyenner ces deux réponses. Ainsi les grandes incertitudes associées à cette situation ont été reconnues et quantifiées. Cette quantification nous éclaire sur l'inadéquation relative du choix gaussien à représenter une situation bimodale. Toutefois en reproduisant par un tirage aléatoire ce comportement, du fait de la grande incertitude, la composante aléatoire remet souvent le robot

face à une situation mieux gérée (obstacle de biais), et l'obstacle se trouve ainsi tout de même évité !

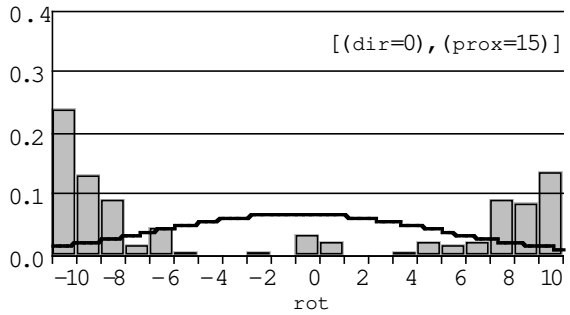


Figure 4-4 — L'effet d'un modèle gaussien sur un phénomène bi-modal.

Pousser les obstacles

Avec le même dispositif, nous télé-opérons à présent le robot en mode APP pour se diriger vers l'obstacle le plus proche et le pousser (il s'agit de cubes de Lego qui peuvent glisser). En mode RES, ce comportement est très bien restitué. Dès qu'on présente au robot un Lego© à la portée de ses capteurs, il se dirige vers lui et le pousse. En fait, il a appris à se diriger simplement à chaque instant dans la direction où les capteurs de proximités sont les plus activés (avec une part d'aléatoire qui, dans ce cas aussi, stabilise en général le comportement). Une fois le comportement bien appris, nous observons que le robot pousse l'obstacle devant lui en ligne droite, avec de temps en temps un léger mouvement de rotation d'un côté ou de l'autre afin de « rattraper » l'obstacle lorsque que ce dernier « glisse » et amorce une rotation.

Au cours de cette expérience nous avons eu l'occasion d'observer un phénomène « émergeant » tout à fait remarquable. Nous avons remarqué que le robot va naturellement se positionner sur l'obstacle de façon à le pousser en ligne droite. Lorsque nous plaçons le Khepera contre une extrémité d'une barre de Lego suffisamment large, la barre pivote sur la gauche et le robot suit. Au bout d'un moment il se retrouve face au centre et pousse tout droit. Si on surcharge le bord de la barre par un poids supplémentaire, le robot se replace naturellement face au centre de pivotement. Le mouvement du robot montre donc des propriétés qui émergent de l'interaction physique avec l'environnement mais qui n'étaient pas prises en compte a priori dans les raisons initiales de le programmer de cette façon. Cette propriété est utilisée dans les travaux de Yoshikawa où un robot détermine un facteur de friction entre le sol et un objet en le poussant et en le faisant glisser [Yoshikawa91].

Les deux films qui suivent montrent comment peut être appris ce comportement. Au début Khepera ne sait rien. À chaque instant (tous les 10es de seconde) il tire au hasard suivant une distribution très plate sa vitesse de rotation. La présence ou l'absence d'obstacle lui est indifférente, ses déplacements sont complètement aléatoires. Le Khepera peut être utilisé suivant les 3 modes présentés précédemment (TEL, APP, RES).

Le premier film montre une succession de ces trois modes. Tout d'abord, le comportement souhaité est démontré en mode télé-opéré. Ensuite vient une succession de phases d'apprentissages et de restitutions. Certaines phases d'apprentissage sont des séquences entières du comportement, d'autres sont très courtes et ont pour but de montrer au robot quoi faire dans telles ou telles situations caractéristiques. Après un total de 26 secondes d'apprentissage, on constate que Khepera sait pousser les obstacles de manière satisfaisante.

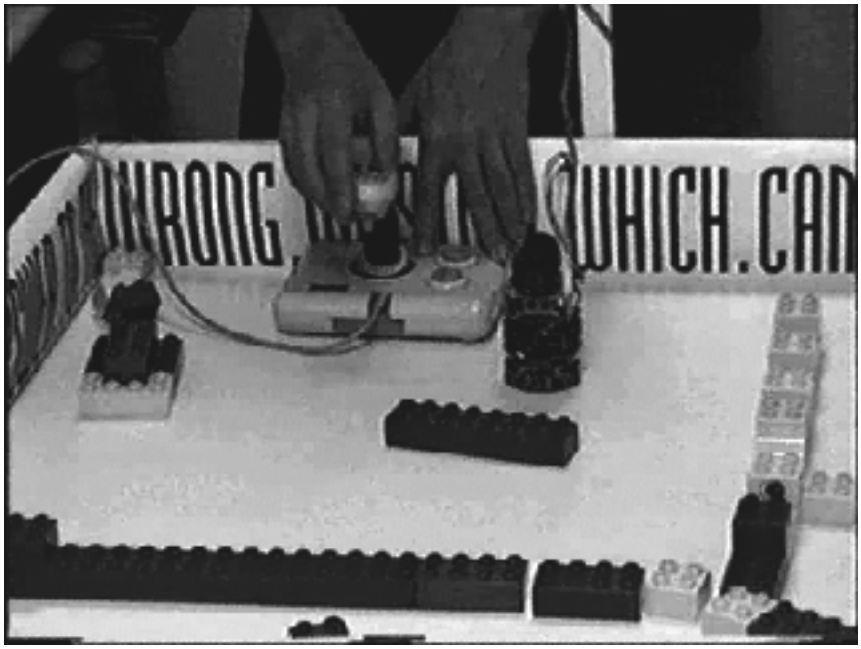


Figure 4-5 — Apprentissage par télé-opération d'un comportement de poussé d'obstacle (film : Film—Chp4a.mov ; durée 2 minutes 3 secondes).

Le deuxième film, tourné sans apprentissage supplémentaire, montre que le Khepera est en mesure de pousser les obstacles indépendamment, de leurs formes, de leurs couleurs ou de leurs natures (dans certaines limites évidemment).

Suivre les contours

Enfin, un comportement de suivi de murs, ou plus généralement de contours, peut également être appris sur la base des mêmes connaissances préalables, pourvu que la télé-opération se fasse toujours dans le même sens (par exemple, toujours garder le contour sur la gauche du robot). Autrement le robot perd souvent le contour quand il arrive à des angles relativement aigus : n'ayant pas de mémorisation des actions précédentes, il ne sait pas dans quel sens tourner pour rattraper le contour, et agit au hasard. Nous reviendrons plus bas sur ce point (section 4.3.3).

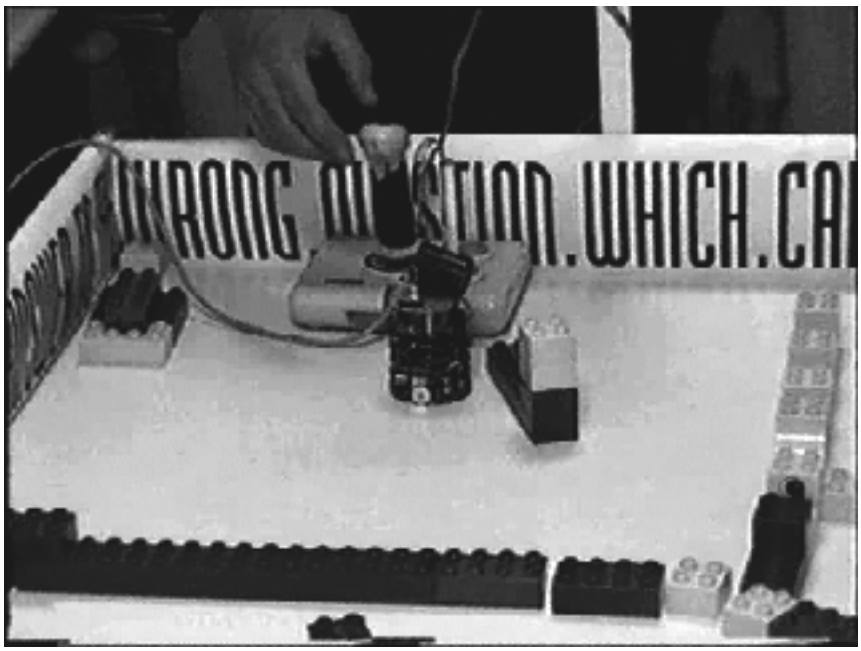


Figure 4-6 — Restitution de l'apprentissage d'un comportement de poussé d'obstacle.
(film : Film—Chp4b.mov ; durée : 35 secondes).

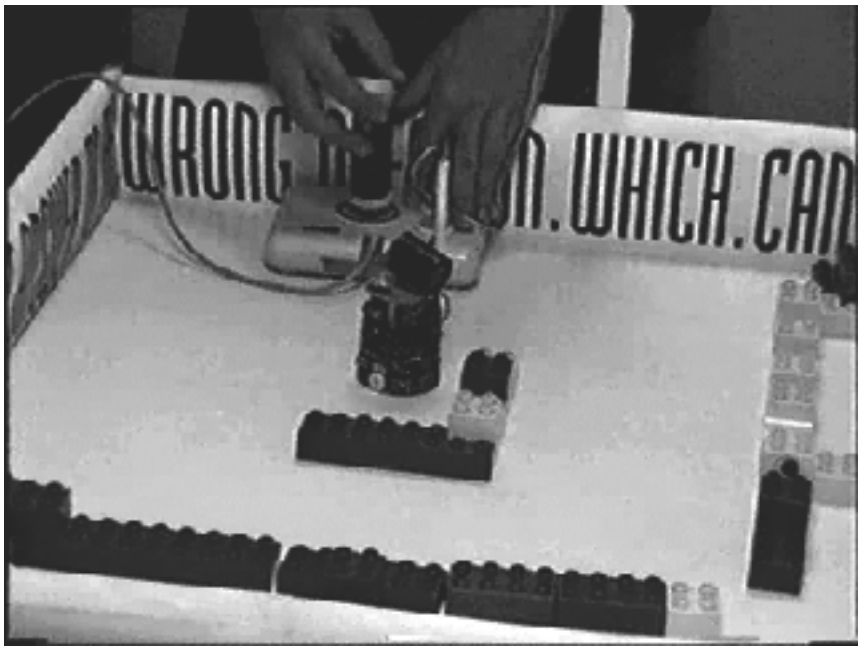


Figure 4-7 — Apprentissage par télé-opération d'un comportement de suivi de contour.
(film :Film—Chp4c.mov ; durée : 1 minute 38 secondes).

4.2.5 Enseignements et discussion

La principale leçon que nous retenons de cette expérience est qu'il est possible d'apprendre des comportements différents à partir des mêmes connaissances préalables en changeant seulement les données servant à identifier les paramètres des distributions.

Cette constatation montre clairement les rôles respectifs que jouent les connaissances préalables et les données expérimentales. Dans le cadre de cette expérience, les connaissances préalables expriment les choix et hypothèses retenus pour modéliser un ensemble de comportements réactifs sous la forme d'une dépendance entre la variable motrice `Vrot` et les informations fournies par les capteurs de proximités (résumées par les variables `Dir` et `Prox`). Ces connaissances décrivent comment un comportement réactif sera perçu du point de vue du robot, c'est-à-dire quelle structure interne au robot sera utilisée pour rendre compte d'un tel comportement. L'aspect fonctionnalité du comportement n'est donc pas spécifié au préalable, c'est par le choix d'un jeu de données spécifique qu'une fonctionnalité particulière sera identifiée.

Par ces expériences, nous illustrons l'importance primordiale que joue l'expérimentation dans ce type de programmation des robots. La même idée sera reprise plus loin dans le cas de tâches de reconnaissance d'objets ou de situations (voir Chapitre 8).

Un autre enseignement important de cette expérience est la démonstration de l'efficacité de ce type d'apprentissage pour les comportements réactifs. Quelques dizaines de secondes d'apprentissage suffisent pour apprendre chacun des trois comportements.

Moins capitale mais néanmoins importante est la preuve apportée par ces expériences que, pour ce type de comportement réactif, les données sensorielles peuvent être drastiquement comprimées sans que cela ne nuise à la qualité de la restitution. On passe pour ces trois expériences d'un espace perceptif de 1024^6 cas possibles (les 1024 valeurs de chacun des 6 capteurs avants de proximité) à un espace de 21x16 cas possibles (les 21 valeurs de `Dir` et les 16 de `Prox`). La représentation probabiliste s'avère un outil intéressant pour tenir compte de ce genre de compression de données. En effet, l'adéquation des pré-traitements choisis peut s'évaluer par l'incertitude sur les distributions obtenues après apprentissage. Si les pré-traitements sont inappropriés ou si la compression des données est trop importante, les distributions obtenues seront très « plates », très incertaines et peu utilisables en restitution car elles conduiront le robot à choisir ce qu'il doit faire totalement au hasard. Un exemple est donné par le modèle gaussien de l'évitement d'un obstacle frontal. Le modèle unimodal n'est pas adéquat pour représenter les données bimodales. Il résulte alors de l'apprentissage une distribution gaussienne ayant un très grand écart-type (voir Figure 4-4). À l'opposé, si les distributions obtenues sont très « pointues » c'est le signe indéniable que les connaissances préalables choisies ont permis d'extraire de l'information des données expérimentales auxquelles elles ont été appliquées. Il existe même une mesure objective de la qualité des distributions obtenues : leur entropie (voir l'article [Bessière98b] inséré en Annexe B pour plus de détail sur la notion d'entropie).

Ces expériences ont aussi montré que le robot, lorsqu'il ne sait pas quoi faire, s'en sort souvent bien en tirant totalement au hasard les valeurs de ces variables motrices. Cette stratégie s'avère efficace car le robot retombe alors en général rapidement dans des situations bien identifiées où il sait efficacement prendre des décisions. Ceci a été rencontré dans le cas de l'évitement frontal d'un obstacle dont il vient d'être question au paragraphe précédent. La distribution gaussienne à grand écart-type identifiée pour cette situation amène le robot à décider à peu près équiprobablement entre un virage à gauche et un virage à droite. À peine ce

virage amorcé le robot se retrouve dans une situation moins ambiguë (obstacle à gauche ou obstacle à droite) où il sait précisément quoi faire pour éviter les obstacles. Des cas similaires sont observés lorsque le robot se trouve dans des situations qu'il n'a jamais rencontrées précédemment. La distribution par défaut (avant apprentissage) étant une gaussienne très plate, il choisit ce qu'il doit faire totalement au hasard et se retrouve alors, en général, rapidement dans une situation connue.

4.3 Spécification a priori de comportements réactifs

4.3.1 Tourner en présence d'obstacles

4.3.1.1 Objectifs et protocole expérimental

Pour les trois expériences précédentes, les paramètres μ_x[`Dir Prox`] et σ_x[`Dir`, `Prox`] de la distribution P(`Vrot` | `Dir Prox` *Cp_Obstacle_Cpt_x*) furent identifiés à partir de données expérimentales obtenues par télé-opération.

L'objectif est ici de définir fonctionnellement les relations μ_r(`Dir Prox`) et σ_r(`Dir Prox`) de manière à obtenir un comportement d'évitement d'obstacle.

4.3.1.2 Spécification

Pour cette expérience, nous décidons que les connaissances préalables seront les mêmes que dans le cas avec apprentissage (section 4.2). Rappelons les pour mémoire.

Choix des variables

Variables pertinentes :

- **`Dir`** : domaine {-10, -9,…, +10}, $k_{\texttt{Dir}} = 21$;
- **`Prox`** : domaine {0, 1,…, 15}, $k_{\texttt{Prox}} = 16$;
- **`Vrot`** : domaine {-10, -9,…, +10}, $k_{\texttt{Vrot}} = 21$.

Connaissances préalables de dépendances

$$\begin{aligned} &\mathrm{P}(\texttt{Prox Dir Vrot} \mid D\ C) \\ &= \mathrm{P}(\texttt{Prox} \mid D\ C)\,\mathrm{P}(\texttt{Dir} \mid D\ C)\,\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ D\ C). \end{aligned} \tag{4.17}$$

Connaissances préalables d'observation

$$\begin{aligned} &P\left(\left[\texttt{Prox} = \texttt{prox}_i\right] \mid D\ C\right) = P\left(\left[\texttt{Prox} = \texttt{prox}_i\right] \mid C\right) = \mathbf{U}_{k\texttt{Prox}} = \frac{1}{16}, \\ &P\left(\left[\texttt{Dir} = \texttt{dir}_j\right] \mid D\ C\right) = P\left(\left[\texttt{Dir} = \texttt{dir}_j\right] \mid C\right) = \mathbf{U}_{k\texttt{Dir}} = \frac{1}{21}, \end{aligned} \tag{4.18}$$

$$\begin{aligned} &P\left(\texttt{Vrot} \mid \left[\texttt{Prox} = \texttt{prox}_i\right]\left[\texttt{Dir} = \texttt{dir}_j\right] D\ C\right) \\ &= \mathbf{G}_{\mu_r(\texttt{proxi},\texttt{dirj}),\ \sigma_r(\texttt{proxi},\texttt{dirj})}(\texttt{Vrot}). \end{aligned} \tag{4.19}$$

4.3.1.3 Identification

Le comportement souhaité est le suivant :

Plus les obstacles sont loin du robot moins il est contraint sur sa vitesse de rotation. Et réciproquement plus il est proche d'un obstacle plus sa vitesse de rotation est contrainte.
Plus l'obstacle se trouve face au robot plus sa vitesse de rotation doit être importante, et inversement. Si l'obstacle se trouve par exemple sur sa gauche alors le robot doit tourner sur la droite mais pas nécessairement avec une vitesse élevée.
Nous avons spécifié les écart-types (qui, rappelons le, traduisent ici l'importance de la contrainte) en fonction uniquement de la valeur de `Prox`. Loin des obstacles, l'écart-type est relativement important (valeur 6,0) ; en se rapprochant des obstacles, il diminue pour atteindre une valeur minimale de 0,4. La décroissance se fait selon une fonction sigmoïde. La sigmoïde est une fonction à deux paramètres α et β, notée $\text{sigmoid}_{\alpha,\beta}$, définie de la manière suivante :

$$\text{sigmoid}_{\alpha,\beta}(x) = \frac{1}{1 + e^{-4\beta(x-\alpha)}}. \tag{4.20}$$

Cette fonction varie entre 0 et 1. Le point d'inflexion de la sigmoïde se situe au point α, et à ce point la pente vaut β.

Nous plaçons le point d'inflexion à `Prox` = 5 ; en ce point, la pente de la sigmoïde vaut 0,2. Rapportant l'échelle de [0, 1] à [0,4, 6,0] nous obtenons la formule (4.21) pour σ_r, dont la courbe apparaît Figure 4-8 :

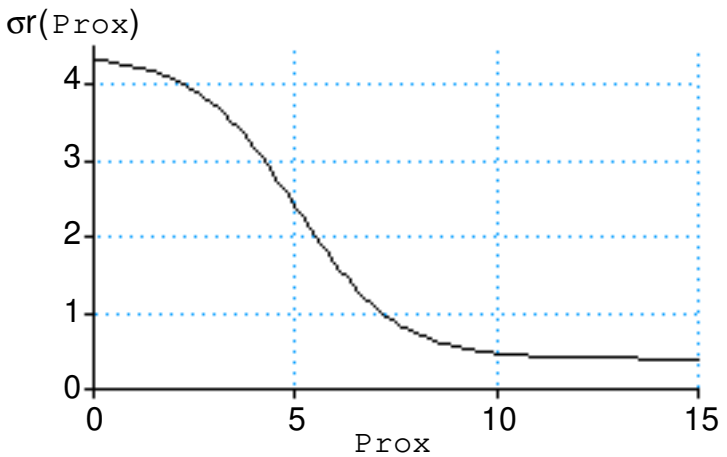


Figure 4-8 — La fonction sigmoïde σ_r(`Prox`).

$$\sigma(\texttt{Prox}) = 6 - 5{,}6 * \text{sigmoid}_{\alpha,\beta}(\texttt{Prox}) \quad \text{avec } \alpha = 5 \text{ et } \beta = 0{,}2. \tag{4.21}$$

La vitesse de rotation doit être négative pour les obstacle à droite (Dir > 0) et positive pour les obstacles à gauche (`Dir` < 0). Pour le cas particulier ou l'obstacle est frontal (`Dir` = 0) nous avons choisi (arbitrairement) de le traiter comme un obstacle à gauche.

Loin des obstacles, la vitesse de rotation tends vers 0. En se rapprochant des obstacles, sa valeur absolue augmente en suivant une fonction sigmoïde en `Prox` et une fonction linéaire en `Dir`. La valeur maximale sera `Vrot` = 9, atteinte pour `Dir` = 0. Nous avons fixé le point d'inflexion de la sigmoïde à `Prox` = 7, la pente valant 0,2 en ce point. La fonction obtenue est la suivante, représentée Figure 4-9 :

$$\mu(\texttt{Dir},\texttt{Prox}) = \text{signe}(\texttt{Dir})\left(\text{sigmoid}_{7,\ 0{,}2}(\texttt{Prox})\right)\left(-2 - 7\left(\frac{10 - |\texttt{Dir}|}{10}\right)\right)$$
$$\text{signe}(\texttt{Dir}) = \begin{cases} -1 \text{ pour } \texttt{Dir} \le 0 \\ 1 \text{ sinon.} \end{cases} \tag{4.22}$$

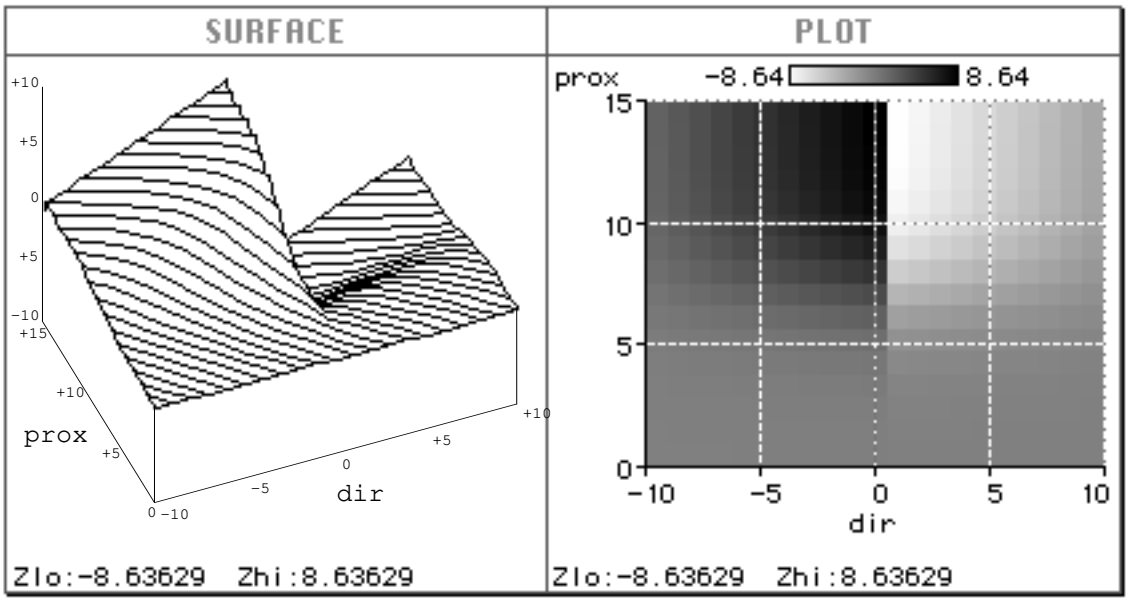


Figure 4-9 — La fonction μ_r(`Dir`, `Prox`).
Cette fonction spécifie l'espérance de la vitesse de rotation pour éviter un obstacle. On remarque la discontinuité autour de la situation « obstacle frontal ». Plus le robot est proche de l'obstacle, plus cette discontinuité est forte car il doit choisir entre un évitement brutal à gauche ou à droite. De plus, plus il est proche et plus la rotation sera fortement imposée (faible écart-type).

Les figures suivantes (Figure 4-10 et Figure 4-11) montrent respectivement comment la probabilité finale P(`Vrot` | `Dir Prox`) se comporte lorsque `Dir` = 7 (obstacle nettement sur la droite) pour différentes valeurs de `Prox`, et comment elle se comporte en fonction de `Dir` lorsque `Prox` = 15 (saturation, donc obstacle très proche) :

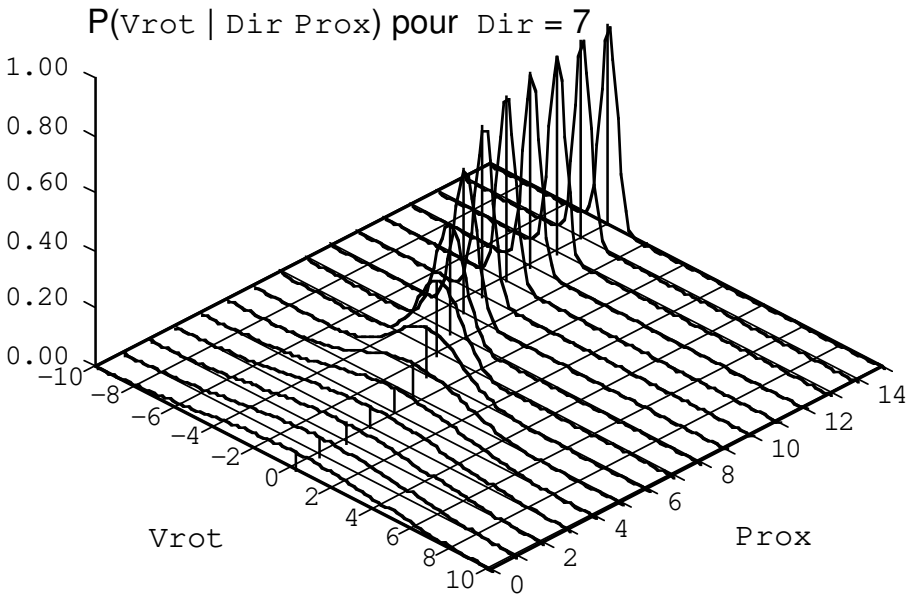


Figure 4-10 — P(Vrot | Dir Prox) pour Dir=7 et différentes valeurs de Prox.
On remarque que plus l'obstacle est éloigné du robot (faible valeur de Prox), moins les contraintes sur la vitesse de rotation sont importantes (la valeur de l'écart type augmente). Lorsque le robot n'est plus en présence d'obstacle (Prox=0), la distribution sur la vitesse de rotation s'assimile à une distribution uniforme.

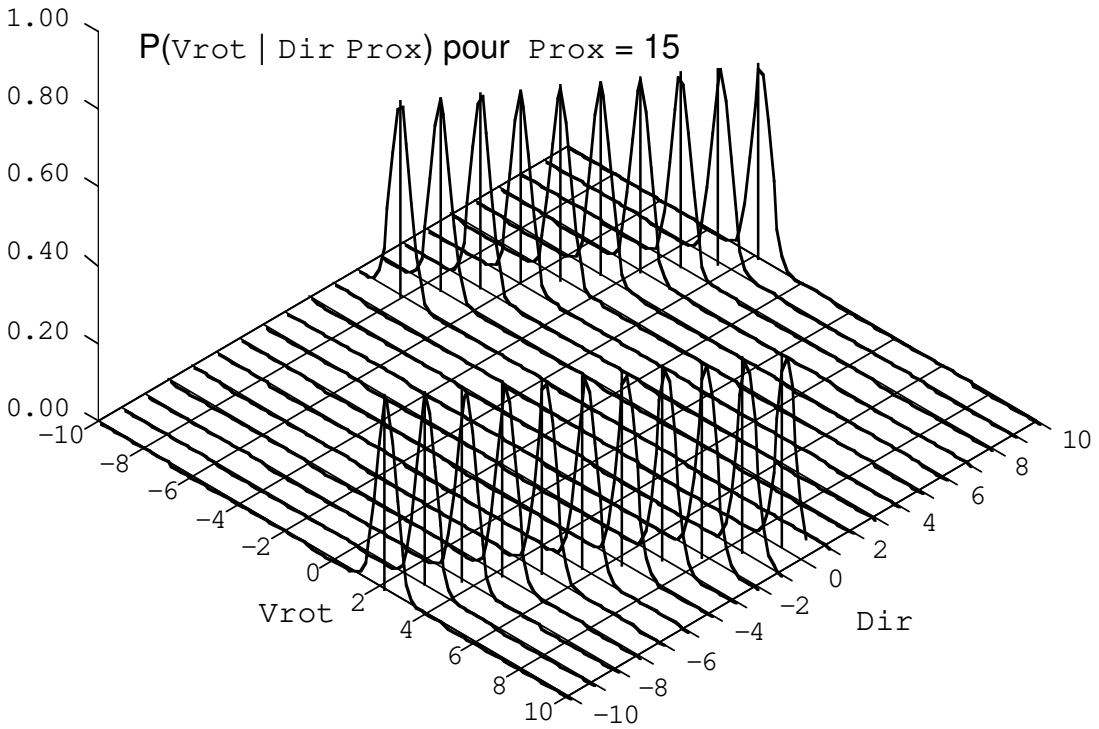


Figure 4-11 — P(Vrot | Dir Prox) pour Prox=15 et différentes valeurs de Dir.
Nous pouvons remarquer à l'aide de cette figure l'apparition d'une discontinuité en présence d'un obstacle frontal (Dir = 0). Ici, nous avons décidé de traiter un obstacle se présentant face au Khepera en l'évitant par la droite (Vrot fortement positif).

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_Tourner* :

$$\begin{aligned} &\mathrm{P}(\texttt{Prox Dir Vrot} \mid \mathit{Cp_Tourner}) \\ &= \mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Tourner})\,\mathrm{P}(\texttt{Dir} \mid \mathit{Cp_Tourner})\,\mathrm{P}(\texttt{Vrot} \mid \texttt{Prox Dir}\ \mathit{Cp_Tourner}). \end{aligned} \tag{4.23}$$

4.3.1.4 Utilisation

Programme et question probabiliste

Le programme est quasiment le même que celui employé dans le cas de l'évitement d'obstacle avec apprentissage (seuls changent, dans la requête, le nom de la connaissance préalable et le fait qu'il n'est plus nécessaire de spécifier de jeu de données) :

1/ Description utilisée :

Cp_Tourner

2/ Requête :

E_q : {Vrot}

E_c : {Dir, Prox}

E_i : Ø

Question : P(Vrot | Dir Prox *Cp_Tourner*)

3/ Décision :

Tirage aléatoire

Résultat

Les résultats de cette expérience sont très satisfaisants pour des vitesses de translation relativement faibles (Vtrans < 5). Le robot évite les obstacles indépendamment de leurs formes et de leurs orientations.

Pour des vitesses de translations supérieures, l'évitement reste globalement correct, toutefois le robot va plus souvent toucher les obstacles, en particulier ceux dont la couleur absorbe l'infrarouge. Comme nous l'avions suggéré plus haut, une solution est de limiter la vitesse en translation en fonction de la proximité des obstacles ; sa formulation probabiliste sera présentée à la section 4.3.2.

Loin des obstacles, le robot a un comportement aléatoire, ce qui correspond bien aux contraintes que nous nous étions fixées. L'objectif de ce comportement est d'éviter les obstacles ; en leur absence, le robot n'a aucune raison d'être contraint outre mesure. Cette « liberté » va se révéler très appropriée dans le cadre de la combinaison de comportement ; voir le Chapitre 6.

4.3.1.5 Enseignements et discussion

Tout d'abord, il est important de noter que les pré-traitements (Dir et Prox) ont été obtenus empiriquement, et n'ont pas nécessité de modèles précis des capteurs. Cependant le

comportement obtenu c'est révélé robuste à des variations significative de l'environnement (changement de la nature et de la mobilité des obstacles, des conditions d'éclairage, etc.). L'utilisation des probabilités autorise donc une démarche empirique car elle permet de définir des modèles avec des degrés variés de précision. Cette démarche empirique est à rapprocher par exemple de celle de Horswill ([Horswill93]) en vision par ordinateur appliquée à la robotique.

Ceci étant dit, la simplicité a son prix. Nous avons interprété `Dir` et `Prox` comme étant la direction et la proximité de l'obstacle le plus proche. Tout se passe bien en présence d'un seul obstacle. Mais qu'en est-il en présence de deux obstacles ?

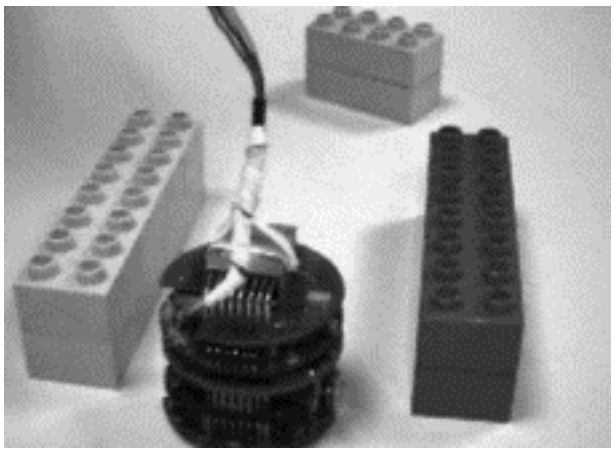


Figure 4-12 — Le Khepera placé devant un couloir.

Dans ce cas, le pré-traitement `Dir` à tendance à indiquer des obstacles en face (`Dir` = 0). En effet, dans la fonction de calcul de `Dir`, les valeurs des capteurs de proximité symétriquement opposés annulent mutuellement leurs effets sur `Dir`. Ceci apparaît clairement lorsque le robot parcours un couloir étroit (Figure 4-12).

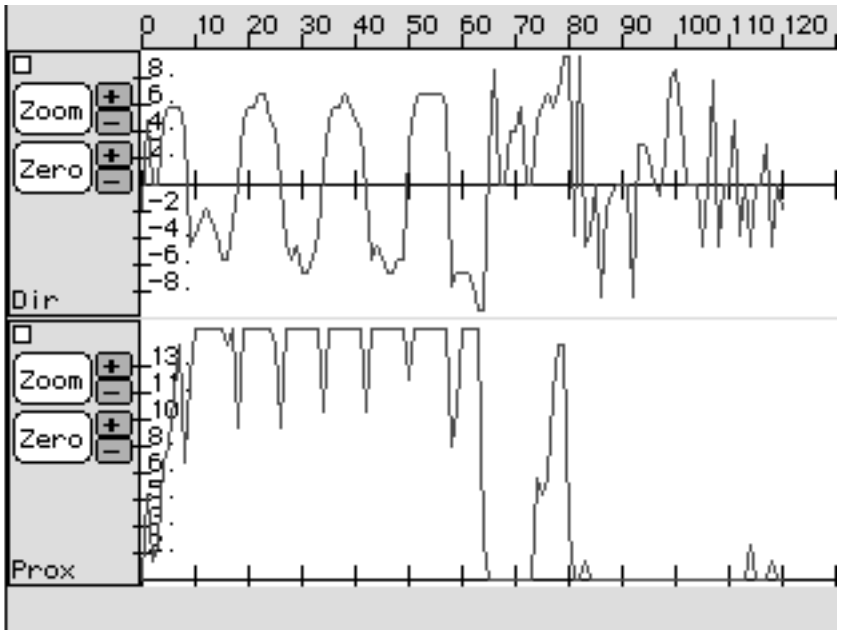


Figure 4-13 — Évolution des variables `Dir` et `Prox` lors du parcours d'un couloir.
Pour réaliser cette séquence nous avons placé le robot à l'entrée d'un couloir. On remarque l'oscillation de `Dir` (couloir). La sortie du couloir intervient à t=65, puis entre t=75 et t=80, le Khepera évite l'obstacle se présentant sur sa droite, pour atteindre l'espace libre.

Dans une telle situation, nous avons observé que le robot adopte un comportement oscillant ; il suit une trajectoire sinusoïdale comme s'il rebondissait sur un mur puis sur l'autre et ainsi de suite. En visualisant `Dir` dans cette situation, nous remarquons que sa valeur varie entre -6 et +6 (Figure 4-13).

Nous devons donc voir ces deux pré-traitements comme un choix de modélisation de l'évitement d'obstacle, qui, comme toute modélisation, a ses conditions de validité. Ce point est rarement cité et reconnu dans la littérature. Notre choix ici a consisté à ne considérer qu'un seul obstacle à la fois : le plus proche. Si le comportement observé semble améliorable dans certaines situations (couloir), ce n'en est pas moins une forme d'évitement d'obstacle.

4.3.2 Ralentir en présence d'obstacle

4.3.2.1 Objectifs et protocole expérimental

Comme nous l'avons remarqué plus haut, le comportement d'évitement d'obstacle est satisfaisant pour des vitesses de translation faible. Toutefois, au-delà d'une certaine limite (`Vtrans` > 5), et en présence d'un environnement très encombré nous observons que le Khepera à tendance à rentrer en contact avec les obstacles. Nous proposons ici une description permettant de remédier à cet inconvénient. L'objectif de cette description est de pouvoir spécifier une consigne de vitesse de translation que le Khepera doit suivre et de contraindre le Khepera à diminuer sa vitesse de translation en présence d'obstacle. Lorsque la proximité des obstacles est trop grande nous souhaitons que la vitesse de translation du Khepera soit nulle.

4.3.2.2 Spécification

Choix des variables

Nous ne retenons que trois variables pour cette description, la variable associée à la vitesse de translation du Khepera (`Vtrans`), une variable consigne (`Vtrans_c`) nous permettant de spécifier une vitesse de translation et enfin la variable `Prox` fournissant au robot une information relative à la proximité des obstacles autour de lui. Nous limitons le domaine de variation de la consigne entre 0 et + 10 de manière à interdire une consigne négative.

Variables pertinentes :

- **`Prox`** : domaine de valeur $\{0, 1,\ldots, 15\}$, $k_{\texttt{Prox}} = 16$;
- **`Vtrans`** : domaine de valeur $\{-10, -9,\ldots, +10\}$, $k_{\texttt{Vtrans}} = 21$;
- **`Vtrans_c`** : domaine de valeur $\{0, 1,\ldots, +10\}$, $k_{\texttt{Vtrans_c}} = 11$.

Connaissances préalables de dépendance

La description du comportement porte sur les variables `Prox`, `Vtrans_c` et `Vtrans`, et correspond à la distribution conjointe :

$$\mathrm{P}\left(\texttt{Prox Vtrans_c Vtrans} \mid D\ C\right). \tag{4.24}$$

Nous allons ici exprimer des connaissances nous permettant de décomposer cette distribution conjointe en un produit de distributions élémentaires. En appliquant la règle du produit **[R1]** nous décidons de décomposer (4.24) de la manière suivante :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Prox Vtrans_c Vtrans} \mid D\ C\right)\\&= \mathrm{P}\left(\texttt{Prox Vtrans_c} \mid D\ C\right)\mathrm{P}\left(\texttt{Vtrans} \mid \texttt{Prox Vtrans_c}\ D\ C\right).\end{aligned} \tag{4.25}$$

Nous pensons qu'il n'existe pas a priori de dépendance entre la proximité d'un obstacle et la vitesse de translation consigne fixé par l'opérateur. Nous traduisons cette hypothèse en spécifiant que les variables `Prox` et `Vtrans_c` sont indépendantes :

$$\mathrm{P}\left(\texttt{Prox Vtrans_c} \mid D\ C\right) = \mathrm{P}\left(\texttt{Prox} \mid D\ C\right)\mathrm{P}\left(\texttt{Vtrans_c} \mid D\ C\right). \tag{4.26}$$

Nous obtenons finalement la décomposition suivante :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Prox Vtrans_c Vtrans} \mid D\ C\right)\\&= \mathrm{P}\left(\texttt{Prox} \mid D\ C\right)\mathrm{P}\left(\texttt{Vtrans_c} \mid D\ C\right)\mathrm{P}\left(\texttt{Vtrans} \mid \texttt{Prox Vtrans_c}\ D\ C\right).\end{aligned} \tag{4.27}$$

Connaissances préalables d'observation

Finalement, il reste à indiquer la forme paramétrique associée à chacune des distributions élémentaires.

Nous ne privilégions pas certaines valeurs de proximité vis-à-vis des autres et nous ne voulons pas favoriser certaines consignes de vitesse en translation ; de plus, dans le cadre de cette expérience nous ne chercherons pas à faire évoluer leurs probabilités. Nous traduisons ces choix par :

$$\begin{aligned}&\mathrm{P}\left(\left[\texttt{Prox} = \texttt{prox}_i\right] \mid D\ C\right)\\&= \mathrm{P}\left(\left[\texttt{Prox} = \texttt{prox}_i\right] \mid C\right) = \mathbf{U}_{k\texttt{Prox}} = \frac{1}{16};\\&\\&\mathrm{P}\left(\left[\texttt{Vtrans_c} = \texttt{vtrans_c}_j\right] \mid D\ C\right)\\&= \mathrm{P}\left(\left[\texttt{Vtrans_c} = \texttt{vtrans_c}_j\right] \mid C\right) = \mathbf{U}_{k\texttt{Vtrans_c}} = \frac{1}{11}.\end{aligned} \tag{4.28}$$

Enfin, nous choisissons de représenter la probabilité de `Vtrans` conditionnée par `Prox` et `Vtrans_c` sous la forme d'un ensemble de lois normales paramétrées par les valeurs par `Prox` et `Vtrans_c` :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Vtrans} \mid \left[\texttt{Prox} = \texttt{prox}_i\right]\left[\texttt{Vtrans_c} = \texttt{vtrans_c}_j\right] D\ C\ \right)\\&= \mathbf{G}_{\mu(\texttt{proxi, vtrans_cj}),\ \sigma(\texttt{proxi, vtrans_cj})}(\texttt{Vtrans}).\end{aligned} \tag{4.29}$$

4.3.2.3 Identification

Pour la distribution P(Vtrans | Prox Vtrans_c), nous avons spécifié la fonction μ_t(Prox, Vtrans_c) à l'aide d'une fonction sigmoïde. La Figure 4-14 présente l'aspect de μ_t(Prox, Vtrans_c) pour une vitesse consigne de 7. Nous avons fixé un écart-type de 0,5 quelles que soient les valeurs de Prox et de Vtrans_c :

$$\sigma_t(\texttt{Prox},\texttt{Vtrans_c}) = 0,5$$
$$\mu_t(\texttt{Prox},\texttt{Vtrans_c}) = \texttt{Vtrans_c}\left(1,0 - \text{sigmoid}_{\alpha,\beta}(\texttt{Prox})\right) \quad (4.30)$$
$$\text{avec } \alpha = 9 \text{ et } \beta = 0,2.$$

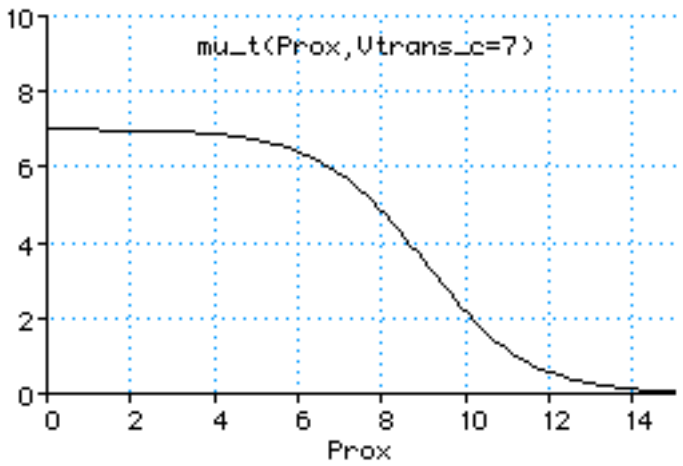


Figure 4-14 — Évolution de μ_t(Prox,Vtrans_c) pour Vtrans_c = 7.

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_Ralentir* :

$$\begin{aligned} &\text{P}\left(\texttt{Prox Vtrans_c Vtrans} \mid Cp_Ralentir\right) \\ &= \begin{vmatrix} \text{P}\left(\texttt{Prox} \mid Cp_Ralentir\right)\text{P}\left(\texttt{Vtrans_c} \mid Cp_Ralentir\right) \\ \text{P}\left(\texttt{Vtrans} \mid \texttt{Prox Vtrans_c } Cp_Ralentir\right). \end{vmatrix} \end{aligned} \quad (4.31)$$

4.3.2.4 Utilisation

Programme et question probabiliste

L'utilisation de cette description consiste à déterminer la vitesse de translation connaissant la valeur de la vitesse consigne et la proximité des obstacles. Ce qui correspond à la requête suivante :

1/ Description utilisée :

Cp_Ralentir

2/ Question :

E_q : {Vtrans}

E_c : {Prox, Vtrans_c}

E_i : Ø

Question : P(Vtrans | Prox Vtrans_c *Cp_Ralentir*)

3/ Décision :

Tirage aléatoire

Résultat

Les résultats obtenus sont satisfaisants. Loin des obstacles, le Khepera a une vitesse de translation correspondant à la consigne. Lorsque la proximité des obstacles augmente, le robot ralentit jusqu'à atteindre une vitesse de translation nulle en présence d'obstacles très proches.

Lors de l'utilisation de cette description, nous avons remarqué qu'en présence d'un obstacle sur le coté le robot ralentit bien que le risque de collision soit moindre qu'avec des obstacles de face. Ceci s'explique par le choix de contraindre la vitesse de translation en ne prenant en compte que la valeur de `Prox`. Pour y remédier, il serait nécessaire de prendre en compte l'information de direction fournit par la variable `Dir` (pour un obstacle latéral il y a moins de contrainte sur la vitesse de translation que pour un obstacle frontal [Laumond90]).

4.3.2.5 Enseignement et discussion

Cette expérience illustre l'utilisation d'une variable particulière, `Vtrans_c`, qui n'est pas à proprement parlé une variable sensorielle du robot. Cette variable nous permet de modifier la consigne sur la vitesse de translation, et, à ce titre, elle est utilisée comme une variable sensorielle vue comme un moyen permettant de fournir de l'information au robot.

En spécifiant cette description, nous avons choisi de contrôler la vitesse de translation indépendamment de la vitesse de rotation. En pratique, nous avons utilisé cette description en l'associant à d'autres descriptions contrôlant la vitesse de rotation. Nous illustrons ce point dans le chapitre 6, présentant l'évitement d'obstacle comme la combinaison de cette description et de la description « tourner en présence d'obstacle » définie plus haut.

4.3.3 Suivi de contour

4.3.3.1 Objectifs et protocole expérimental

L'objectif de cette expérience est de spécifier un comportement de suivi de contours : étant donné un objet de forme et de taille quelconque (ou un mur), le robot suit un contour en restant à une distance constante de l'objet.

Cette expérience illustre un autre aspect des modèles probabilistes. Nous allons commencer par programmer le suivi de contour par un asservissement ne présentant, pour une

fois, aucune incertitude. Cet asservissement sera fonction des capteurs de proximités qui n'auront subi aucun pré-traitement. Cependant, par la suite nous désirerons fonder le même réflexe sur d'autres pré-traitements de type `Dir` et `Prox`, moins intuitivement exploitables pour le suivi de contour mais présentant certains avantages par ailleurs. Pour cela, nous traduirons automatiquement le premier modèle (asservissement) sur les nouveaux pré-traitements. C'est au cours de cette traduction que des incertitudes apparaîtront, dûes à une perte d'information. L'utilisation d'un modèle probabiliste permettra alors à ce niveau de quantifier et de tenir compte de ces incertitudes.

Avant d'attaquer le cœur du sujet, toutefois, il nous faut expliquer pourquoi il s'est avéré nécessaire de programmer en fait non pas un mais deux comportements (symétriques) : suivi de contour à gauche et suivi de contour à droite.

Séparation en suivi par la gauche et suivi par la droite

Lorsque le Khepera qui suit un contour arrive à un angle vif, il peut arriver qu'il « perde de vue » l'objet sur ses capteurs de proximité. Doit-il, alors, le retrouver sur la droite ou la gauche ?

Une solution serait d'avoir une certaine « mémoire » des actions passées : s'il était en train de suivre un objet sur la gauche, ayant perdu l'objet il tournerait à gauche pour le retrouver, et et inversement pour un suivi sur la droite. Mais il y aurait alors un problème dans le cas d'un couloir. Dans un couloir, le robot suit-il les contours sur la droite ou sur la gauche ? Dans quel sens faut-il alors rattraper le contour s'il le perd de vue (situation de la Figure 4-15) ?

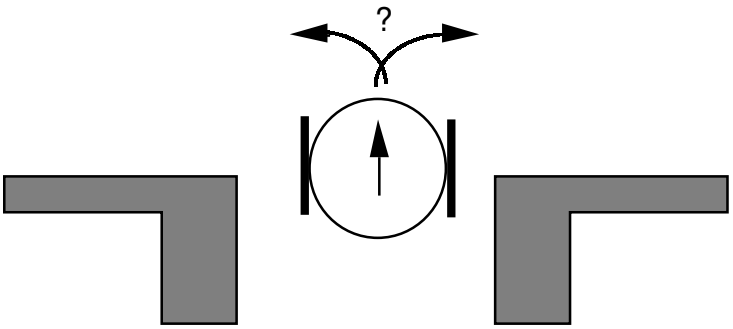


Figure 4-15 — Robot débouchant d'un couloir.

Nous avons remarqué auparavant cette difficulté dans les expériences d'apprentissage par télé-opération. En effet, il est apparu impossible d'apprendre à la fois un comportement de suivi par la gauche et par la droite. La raison principale est très simple, et nous est apparue en observant le robot lors des phases de restitutions de l'apprentissage. Faire un suivi de contour consiste à effectuer un mouvement de rotation permettant de s'éloigner du mur lorsque le robot est trop proche ; et inverser cette rotation lorsqu'il s'en éloigne de trop. Ainsi, lorsqu'il est placé dans un espace libre, très loin d'une surface à longer, le Khepera s'obstine à tourner toujours dans le même sens, sur la droite pour un suivi de contour par la droite et inversement pour un suivi de contour par la gauche.

En étudiant ce problème, nous avons essayé d'intégrer dans le temps les valeurs de `Dir`, mais, dans un couloir, elles s'annulaient à nouveau entre gauche et droite (phénomène d'oscillations brutales entre valeurs 6 et –6, voir Figure 4-13), et le problème n'était pas résolu.

En définitive, un comportement unique de suivi de contour semble inévitablement nécessiter, dans certains cas de sortie de couloir ou de situation « encombrée », une stratégie de décision prise à un plus haut niveau de contrôle.

Nous avons alors préféré assumer dès le début cet aspect des choses et avoir deux comportements séparés pour le suivi par la droite et le suivi par la gauche. De plus cela permet, pour ces comportements élémentaires, de conserver un aspect purement réflexe, c'est-à-dire qui ne nécessite pas une mémoire des états passés.

Dans l'exemple qui suit nous ne traitons que du suivi par la gauche (le suivi par la droite est son symétrique). Nous ne considèrerons alors que les informations sensorielles relatives au côté gauche.

Réalisation d'un asservissement

Un suivi de contour peut être vu comme une tâche d'asservissement. L'objectif est de se maintenir à une certaine distance du contour. Nous avons, dans un premier temps, représenté ce comportement sous la forme d'un asservissement de type proportionnel. Nous fixons des consignes pour les capteurs de proximités `Px0` `Px1` `Px2` et déterminons la vitesse de rotation proportionnellement à l'erreur mesurée entre les consignes et les valeurs courante des trois capteurs. Les valeurs des consignes ont été obtenues en plaçant le robot dans la situation souhaitée et en mesurant la valeur des capteurs de proximité. Les différents gains ont été ajustés empiriquement. La formule obtenue est la suivante :

$$\begin{aligned}
&\text{Vrot} = \text{g}(\texttt{Px0},\texttt{Px1},\texttt{Px2}) \\
&\quad = 8\left(G_{r0}\left(\texttt{Px0}^{*} - \texttt{Px0}\right) + G_{r1}\left(\texttt{Px1}^{*} - \texttt{Px1}\right) + G_{r2}\texttt{Px2}\right) \\
&\texttt{Px0}^{*} = 220 \ : \text{ consigne sur } \texttt{Px0} \\
&\texttt{Px1}^{*} = 550 \ : \text{ consigne sur } \texttt{Px1} \\
&G_{r0} = 0{,}0005 \ : \text{ gain appliqué à l' erreur sur } \texttt{Px0} \\
&G_{r1} = 0{,}0003 \ : \text{ gain appliqué à l' erreur sur } \texttt{Px1} \\
&G_{r2} = 0{,}002 \ \ : \text{ gain appliqué à } \texttt{Px2}\,.
\end{aligned} \tag{4.32}$$

Nous jouons également sur la vitesse de translation, afin d'amortir le comportement de suivi (réduire la vitesse de translation lorsque le robot est loin des consignes à un effet régulateur et permet d'obtenir un suivi plus fin). Le principe consiste à limiter la vitesse de translation, ici une vitesse consigne spécifiée par l'utilisateur et notée `Vtrans_c`, proportionnellement à la valeur absolue de l'erreur :

$$\begin{aligned}
&\text{Vtrans} = \text{h}(\texttt{Px0},\texttt{Px1},\texttt{Px2}) \\
&\quad = \texttt{Vtrans_c}\left(1 - \left(G_{t0}\left|\texttt{Px0}^{*} - \texttt{Px0}\right| + G_{t1}\left|\texttt{Px1}^{*} - \texttt{Px1}\right| + G_{t2}\texttt{Px2}\right)\right) \\
&G_{t0} = 0{,}0011, \ \ G_{t1} = 0{,}0007, \ \ G_{t2} = 0{,}0027 \\
&\texttt{Px0}^{*} = 220, \ \texttt{Px1}^{*} = 550, \\
&\texttt{Vtrans_c} \ : \text{ vitesse de translation consigne (8).}
\end{aligned} \tag{4.33}$$

Cet asservissement, spécifié de façon déterministe, a donné les résultats souhaités.

Cependant, une telle approche de type « asservissement » ne convient pas à toutes les variables. Si par exemple nous cherchons à utiliser `Dir` et `Prox` en considérant comme consignes leurs valeurs mesurées dans la situation de référence, l'approche « asservissement » peut s'avérer plus difficile à mettre en œuvre (`Dir` et `Prox` étant fortement discrétisées et intégrant globalement les informations spatiales, elles ne varient pas de façon continue et

monotone avec Vrot). C'est seulement avec Px0, Px1 et Px2 que nous avons su imaginer facilement un réflexe adéquat.

Malgré cela on peut néanmoins préférer utiliser les variables Dir et Prox pour de multiples raisons, notamment la baisse de la dimension du problème et l'utilisation de ressources génériques réutilisables par ailleurs.

Nous montrons à présent comment il est possible, dans le cadre probabiliste, de changer ainsi de pré-traitements en préservant le comportement. Le prix à payer pour la compression d'information résultant de l'utilisation de pré-traitements est l'apparition d'incertitudes et donc d'une composante aléatoire.

4.3.3.2 Spécification

Choix des variables

Nous n'utilisons pas en fait Dir et Prox, mais des versions « latéralisées » DirG et ProxG qui n'exploitent que les capteurs gauches (on pourrait faire de même avec les droits) :

$$\mathtt{DirG} = f1(\mathtt{Px0},\mathtt{Px1},\mathtt{Px2}) = -\mathrm{Floor}\left(\frac{90\mathtt{Px0}+45\mathtt{Px1}+5\mathtt{Px2}}{9(1+\mathtt{Px0}+\mathtt{Px1}+\mathtt{Px2})}\right), \tag{4.34}$$

$$\mathtt{ProxG} = f2(\mathtt{Px0},\mathtt{Px1},\mathtt{Px2}) = \mathrm{Floor}\left(\frac{\max(\mathtt{Px0},\mathtt{Px1},\mathtt{Px2})}{64}\right). \tag{4.35}$$

DirG et ProxG représentent respectivement la direction de l'obstacle gauche et sa proximité. Les valeurs de DirG varient entre -10 et 0 (11 cas possibles), celles de ProxG entre 0 et 15 (16 cas possibles). Notons que la consigne sur Px0 Px1 Px2 (qui leur assigne respectivement 220, 550 et 0) se traduit en DirG = -9 et ProxG = 8.

Ainsi, en appliquant ce pré-traitement, nous projetons un espace sensoriel de dimension 3 (1024^3 cas possibles) dans un espace sensoriel de dimension inférieur, 2 ($16\times11 = 176$ cas possibles).

Nous aurions également pu prendre en compte la variable Vtrans_c en la considérant comme une variable sensorielle au même titre que DirG et ProxG ; mais par simplification nous avons décidé de la fixer à 8 (qui correspond à une vitesse de translation de 64 mm.s^{-1}).

Variables pertinentes :

- **DirG** : domaine de valeur {-10, -9,…, 0} }, $k_{\mathtt{DirG}} = 11$;
- **ProxG** : domaine de valeur {0, 1,…, 15}, $k_{\mathtt{ProxG}} = 16$ cas ;
- **Vrot** : domaine de valeur {-10, -9,…, +10}, $k_{\mathtt{Vrot}} = 21$ cas ;
- **Vtrans** : domaine de valeur {-10, -9,…, +10}, $k_{\mathtt{Vtrans}} = 21$ cas.

Connaissances préalables de dépendance :

La description du comportement porte sur les variables DirG, ProxG, Vrot, Vtrans, et correspond à la distribution conjointe :

$$\mathrm{P}(\texttt{ProxG DirG Vrot Vtrans} \mid D\ C), \tag{4.36}$$

que nous décomposons sous la forme :

$$\mathrm{P}(\texttt{ProxG DirG Vrot Vtrans} \mid D\ C) = \begin{vmatrix} \mathrm{P}(\texttt{ProxG} \mid D\ C) \\ \mathrm{P}(\texttt{DirG} \mid D\ C) \\ \mathrm{P}(\texttt{Vrot} \mid \texttt{ProxG DirG}\ D\ C) \\ \mathrm{P}(\texttt{Vtrans} \mid \texttt{ProxG DirG}\ D\ C) \end{vmatrix}. \tag{4.37}$$

Ceci traduit les hypothèses suivantes : d'une part nous considérons `ProxG` et `DirG` indépendantes ne sachant rien d'autre :

$$\mathrm{P}(\texttt{ProxG DirG} \mid D\ C) = \mathrm{P}(\texttt{ProxG} \mid D\ C)\mathrm{P}(\texttt{DirG} \mid D\ C), \tag{4.38}$$

et d'autre part, conditionnellement à la connaissance de `DirG` et `ProxG`, les variables `Vrot` et `Vtrans` sont indépendantes.

Connaissances préalables d'observation

Nous ne privilégions pas certaines situations sensorielles vis-à-vis des autres, de plus, dans le cadre de cette expérience nous ne chercherons pas à faire évoluer leurs probabilités. Nous traduisons ces choix par :

$$\begin{aligned} &\mathrm{P}([\texttt{ProxG} = \texttt{proxG}_i] \mid D\ C) = \mathrm{P}([\texttt{ProxG} = \texttt{proxG}_i] \mid C) = \mathbf{U}_{k\texttt{ProxG}} = \frac{1}{16}, \\ &\mathrm{P}([\texttt{DirG} = \texttt{dirG}_j] \mid D\ C) = \mathrm{P}([\texttt{DirG} = \texttt{dirG}_j] \mid C) = \mathbf{U}_{k\texttt{DirG}} = \frac{1}{11}. \end{aligned} \tag{4.39}$$

Enfin, nous décidons de représenter les probabilités de `Vrot` et de `Vtrans` conditionnées par `DirG` et `ProxG` sous la forme d'un ensemble de lois de Gauss paramétrées par les valeurs de `DirG` et `ProxG` :

$$\begin{aligned} &\mathrm{P}(\texttt{Vrot} \mid [\texttt{ProxG} = \texttt{proxG}_i][\texttt{DirG} = \texttt{dirG}_j]\ D\ C) \\ &= \mathbf{G}_{\mu r[\texttt{proxGi, dirGj}],\ \sigma r[\texttt{proxGi, dirGj}]}(\texttt{Vrot}), \\ &\mathrm{P}(\texttt{Vtrans} \mid [\texttt{ProxG} = \texttt{proxG}_i][\texttt{DirG} = \texttt{dirG}_j]\ D\ C) \\ &= \mathbf{G}_{\mu t[\texttt{proxGi, dirGj}],\ \sigma t[\texttt{proxGi, dirGj}]}(\texttt{Vtrans}). \end{aligned} \tag{4.40}$$

4.3.3.3 Identification

Il s'agit de déterminer les valeurs des paramètres μ_r, σ_r, μ_t et σ_t en fonction de `DirG` et `ProxG`, connaissant d'une part la spécification de ces deux pré-traitements en fonction de `Px0`, `Px1` et `Px2`, et d'autre part une relation permettant de fixer les commandes `Vrot` et `Vtrans` en fonction de `Px0`, `Px1` et `Px2` (la Figure 4-16 représente les relations entres les

variables pour la commande de la vitesse de rotation). Nous allons détailler la démarche suivie pour la vitesse de rotation `Vrot`, pour `Vtrans`, la méthode suivie est identique.

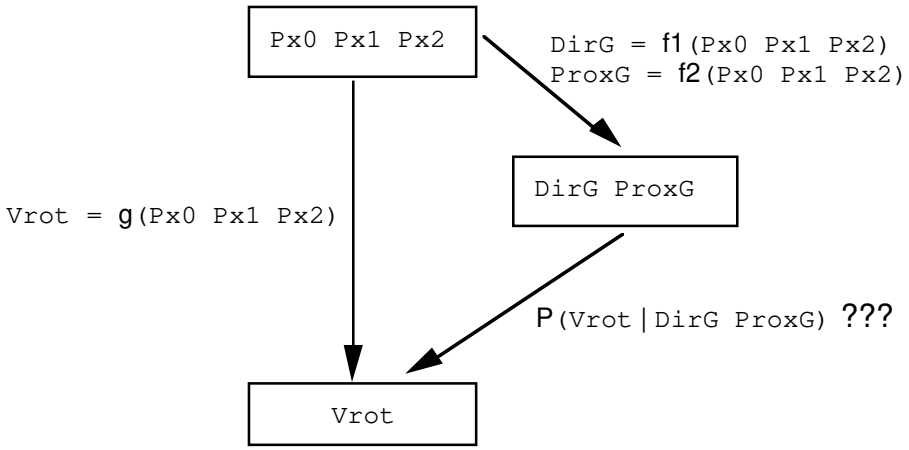


Figure 4-16 — Approximation probabiliste d'une loi de contrôle.
La relation fonctionnelle entre la variable `Vrot` et les variables `Px0 Px1 Px2` (à gauche) est approximée par le biais d'un pré-traitement définissant `DirG` et `ProxG` en fonction de `Px0 Px1 Px2` et d'une description décrivant la dépendance entre `Vrot DirG` et `ProxG`.

L'idée intuitive est très simple. Nous avons choisi de décrire la relation de dépendance entre `Vrot` et les variables `DirG` et `ProxG` selon les connaissances préalables que nous venons d'énoncer plus haut (ensemble de lois de Gauss). Afin d'identifier les paramètres de ces lois nous allons générer « artificiellement », sans le recours à des mesures effectuées avec le robot, un ensemble de donnés formé de triplets (`vrot`, `dirG`, `proxG`). En parcourant tous les cas des variables brutes `Px0 Px1 Px2`, nous pouvons, pour chaque triplet particulier (`px0`, `px1`, `px2`) générer un triplet associé :

(`vrot`, `dirG`, `proxG`) = (g(`px0`, `px1`, `px2`), f1(`px0`, `px1`, `px2`), f2(`px0`, `px1`, `px2`)).

L'algorithme suivant permet, en parcourant chaque valeur de `Px0`, `Px1` et `Px2`, de déterminer le nombre d'occurrences d'un triplet particulier (`vrot`, `dirG`, `proxG`) et le mémoriser dans la table `T_n` :

```
Pour chaque valeur de Px0 Px1 Px2 faire
   vrot ← g(Px0, Px1, Px2)
   dirG ← f1(Px0, Px1, Px2)
   proxG ← f2(Px0, Px1, Px2)
   T_n[dirG, proxG, vrot] ← T_n[dirG, proxG, vrot] + 1
fin Pour
```

On pourrait utiliser le tableau `T_n` directement comme modèle probabiliste (sous la forme d'une loi de succession de Laplace), mais les connaissances préalables que nous avons énoncées plus haut spécifient les distributions P(`Vrot` | `DirG`, `ProxG`) par des gaussiennes, réduisant ainsi le nombre de paramètres et permettant d'obtenir un modèle de même structure que ceux utilisés jusqu'ici. Ceci au prix d'une approximation et d'une augmentation des incertitudes :

```
;; Identification des paramètres μr et σr
Pour chaque valeur de DirG ProxG
    n ← 0
    S1 ← 0
    S2 ← 0
    Pour chaque valeur de Vrot
        n ← n + T_n[DirG, ProxG, Vrot]
        S1 ← T_n[DirG, ProxG, Vrot] + Vrot
        S2 ← T_n[DirG, ProxG, Vrot] + Vrot2
    fin Pour
    μr[DirG, ProxG] ← S1 / n
    _r[DirG, ProxG] ← sqrt(S2/ n – (S1 / n)2)
fin Pour
```

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_Suivi* :

$$\mathrm{P}(\texttt{ProxG DirG Vrot Vtrans} \mid Cp_Suivi) = \begin{vmatrix} \mathrm{P}(\texttt{ProxG} \mid Cp_Suivi) \\ \mathrm{P}(\texttt{DirG} \mid Cp_Suivi) \\ \mathrm{P}(\texttt{Vrot} \mid \texttt{ProxG DirG } Cp_Suivi) \\ \mathrm{P}(\texttt{Vtrans} \mid \texttt{ProxG DirG } Cp_Suivi). \end{vmatrix} \quad (4.41)$$

4.3.3.4 Utilisation

Programme et question probabiliste

L'utilisation de cette description pour un suivi de contour sur la gauche consiste à déterminer les consignes des vitesses en translation et en rotation connaissant la valeur des variables `DirG` et `ProxG`. Ce qui correspond à la requête suivante :

1/ Description utilisée :

Cp_Suivi

2/ Requête :

E_q : {Vrot Vtrans}

E_c : {DirG, ProxG}

E_i : Ø

Question : P(Vrot Vtrans | ProxG DirG *Cp_Suivi*)

3/ Décision :

Tirage aléatoire

Résultat

L'effet de ces transformations successives sur le comportement ne se sent absolument pas. Il semble même à la limite meilleur, en gérant mieux certains cas pathologiques où l'asservissement initial (sans incertitudes) pouvait être pris dans des cycles oscillants ou des minima locaux. Paradoxalement, l'adjonction d'incertitudes à un modèle (on parle parfois « d'injection de bruit ») peut donc le rendre plus performant.

4.3.3.5 Enseignement et discussion

La leçon à retenir de cette expérience est que nous avons là une méthode efficace pour « projeter » un asservissement (ou plus généralement un programme) d'un espace de grande dimension dans un espace plus réduit.

Il existe un moyen simple d'évaluer la qualité de la transformation effectuée : plus la distribution obtenue est « plate » (i.e. incertaine) moins bonne a été la transformation. Cette qualité peut être quantifiée par l'entropie de la distribution résultat. Plus l'entropie de cette distribution est faible, meilleure est le résultat. L'incertitude ainsi mesurée sera l'image de l'adéquation d'une part des choix des pré-traitements appliqués aux variables sensorielles de base (les fonctions f1 et f2) et, d'autres part, des connaissances préalables associées à la description.

Chapitre 5

Modèle capteurs

Jusqu'ici nous avons utilisé des pré-traitements spécifiés fonctionnellement pour déterminer les différentes variables sensorielles utilisées : `Dir`, `Prox`, `Lum`. Nous allons illustrer maintenant comment définir un pré-traitement sous la forme d'une description.

5.1 Objectifs et protocole expérimental

Nous avons défini le comportement phototaxie par une description mettant en relation la vitesse de rotation `Vrot` avec un pré-traitement `Lum` appliqué aux capteurs de luminosité `Lmi`. La variable `Lum` retourne la valeur angulaire correspondant à l'indice de la cellule la plus éclairée. Nous pouvons interpréter cette variable comme la direction relative du robot et d'une source lumineuse présente dans la scène.

L'objectif de cette expérience est d'illustrer comment à partir de modèles capteurs définis sous la forme de description nous sommes en mesure de calculer la direction d'une source lumineuse relativement au robot. Pour cela, nous allons passer par deux étapes :

- La première consiste à construire, pour chaque capteur, une description décrivant le modèle de la réponse du capteur de luminosité en présence d'une source lumineuse dont nous connaissons la direction et la distance par rapport au robot.
- Puis dans une deuxième phase, nous allons fusionner ces modèles associés à chacun des capteurs de luminosité afin de déterminer la direction d'une source lumineuse par rapport au robot.

5.2 Phase 1 : modèle capteur

5.2.1 Spécification

5.2.1.1 Choix des variables

Nous avons retenu trois variables pour construire la description du modèle de la réponse d'un capteur de luminosité en présence d'une source lumineuse :

`Lmi` est la variable associée au capteur de luminosité *i*, la valeur de cette variable varie entre 0 (luminosité forte) et 511 (luminosité faible). Domaine de valeur {0, 1,…, 511} avec un pas de 1 soit 512 cas possibles.

`ThetaL` représente la direction de la source lumineuse par rapport à la direction avant du Khepera. Les valeurs de cette variable varient entre –180° et +180°. Nous avons choisi de discrétiser cette variable avec un pas de 10°. Le domaine de valeur est {-180, -170, …, +170} avec un pas de 10 soit 36 cas possibles.

`DistL` représente la distance de la source lumineuse par rapport au Khepera. Les valeurs de cette variable varient entre 0 cm et 25 cm, discrétisées avec un pas de 1 cm. Le domaine de valeur est {0, 1, …, 25} avec un pas de 1 soit 26 cas possibles.

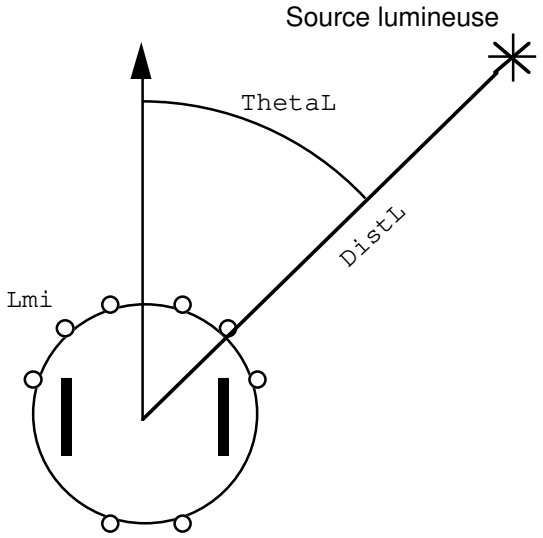


Figure 5-1 — Choix des variables pertinentes pour le modèle de la réponse d'un capteur de luminosité. `Lmi` est la valeur retournée par le capteur de luminosité *i*, `ThetaL`, la direction de la source par rapport au Khepera et `DistL`, la distance de la source par rapport au Khepera.

Variables pertinentes

- **`ThetaL`** : domaine de valeur {-180, -170,…, +170}, $k_{\texttt{ThetaL}} = 36$ cas ;
- **`DistL`** : domaine de valeur {0, 1,…, 25}, $k_{\texttt{DistL}} = 26$ cas ;
- **`Lmi`** : domaine de valeur {0, 1,…, 511}, $k_{\texttt{Lmi}} = 512$ cas.

5.2.1.2 Connaissances préalables de dépendance

La description du modèle porte sur les variables `ThetaL`, `DistL`, `Lmi` et correspond à la distribution conjointe :

$$\mathrm{P}(\texttt{ThetaL DistL Lmi} \mid D\ C). \tag{5.1}$$

Nous décomposons la distribution conjointe sous la forme du produit de trois distributions élémentaires :

$$\begin{aligned}&\mathrm{P}(\texttt{ThetaL DistL Lmi} \mid D\ C)\\ &= \left|\begin{array}{l}\mathrm{P}(\texttt{ThetaL} \mid D\ C)\\ \mathrm{P}(\texttt{DistL} \mid \texttt{ThetaL}\ D\ C)\\ \mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ D\ C).\end{array}\right.\end{aligned} \tag{5.2}$$

Nous considérons que la connaissance de la direction de la source lumineuse n'apporte aucune information sur la distance à laquelle elle se trouve par rapport au robot. Nous traduisons cet a priori en considérant que la probabilité de `DistL` est indépendante de `ThetaL` :

$$\begin{aligned}&\mathrm{P}(\texttt{ThetaL DistL Lmi} \mid D\ C)\\ &= \mathrm{P}(\texttt{ThetaL} \mid D\ C)\mathrm{P}(\texttt{DistL} \mid D\ C)\mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ D\ C).\end{aligned} \tag{5.3}$$

Cette décomposition est très intéressante dans la mesure où elle permet d'exprimer d'une part nos a priori sur la distance et la direction de la source lumineuse par rapport au robot et, d'autre part, une distribution représentant la valeur Lmi de la réponse du capteur de luminosité connaissant les informations relatives à la source lumineuse.

5.2.1.3 Connaissances préalables d'observation

Nous ne privilégions pas certaines positions de la source lumineuse vis-à-vis d'autres (la source lumineuse peut être dans n'importe quelle direction par rapport au robot) et nous ne souhaitons pas observer leur répartition expérimentale :

$$\begin{aligned}&\mathrm{P}([\texttt{ThetaL} = \texttt{thetaL}_i] \mid D\ C) = \mathrm{P}([\texttt{ThetaL} = \texttt{thetaL}_i] \mid C) = \mathbf{U}_{k_{\texttt{ThetaL}}} = \frac{1}{36},\\ &\mathrm{P}([\texttt{DistL} = \texttt{distL}_i] \mid D\ C) = \mathrm{P}([\texttt{DistL} = \texttt{distL}_i] \mid C) = \mathbf{U}_{k_{\texttt{DistL}}} = \frac{1}{26}.\end{aligned} \tag{5.4}$$

Nous choisissons de représenter la probabilité de `Lmi` conditionnée par les valeurs de `ThetaL` et de `DistL` sous la forme d'un ensemble de lois normales :

$$\begin{aligned}&\mathrm{P}(\texttt{Lmi} \mid [\texttt{ThetaL} = \texttt{thetaL}_j]\ [\texttt{DistL} = \texttt{distL}_k]\ D\ C)\\ &= \mathbf{G}_{\mu[\texttt{thetaLj,distLk}],\sigma[\texttt{thetaLj,distLk}]}(\texttt{Lmi}).\end{aligned} \tag{5.5}$$

5.2.2 Identification des paramètres

La phase d'identification des paramètres consiste à déterminer pour chaque situation définie par les valeurs que peuvent prendre les variables `ThetaL` et `DistL` la valeur de la

moyenne μ de la luminosité mesurée et son écart-type σ. En nous inspirant des courbes caractéristiques des capteurs de luminosité fournies avec la documentation du Khepera [K-Team94], nous avons défini une fonction permettant de calculer la valeur moyenne μ en fonction de l'orientation et la distance de la source lumineuse et la position angulaire du capteur de luminosité θ_i. : $\mu\mathbf{LM}(\texttt{ThetaL}, \texttt{DistL}, \theta_i)$. Nous avons fixé l'écart-type à une valeur fixe (ici 20 unités de luminosité) indépendamment de la position de la source lumineuse. La fonction $\mu\mathbf{LM}$ est définie de la façon suivante :

$$\begin{aligned}&\mu\mathrm{LM}(\texttt{ThetaL},\texttt{DistL},\theta_i)\\&= 511 - (\mathrm{K}(\texttt{ThetaL},\theta_i)\mathrm{LM}(\texttt{DistL})).\end{aligned} \tag{5.6}$$

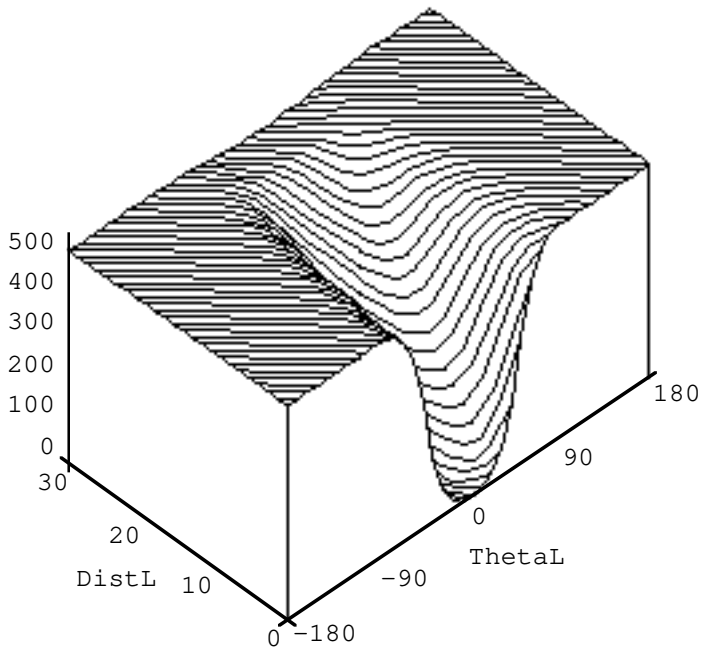


Figure 5-2 — Profil de la réponse d'un capteur de luminosité
Ce profil est déterminé en fonction de la distance (DistL) et de l'orientation (ThetaL) de la source lumineuse.

La fonction **LM** représente la quantité de lumière reçue par le capteur en fonction de la distance du capteur avec la source lumineuse, approximée par une décroissance linéaire entre 5 et 25 cm :

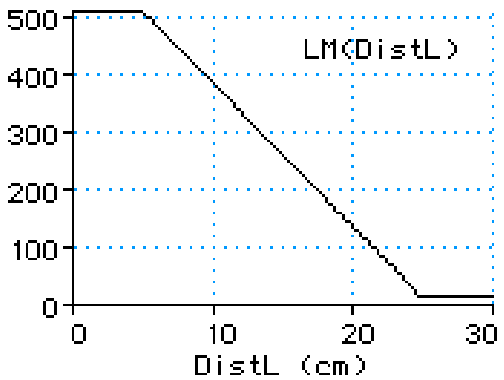


Figure 5-3 — Fonction **LM**(DistL).
Cette fonction représente la quantité de luminosité reçue par le capteur en fonction de la distance du capteur à la source lumineuse.

$$\mathrm{LM}(\texttt{DistL}) = \begin{cases} 511 \text{ si } \texttt{DistL} \ \le 5 \\ 15 \ \text{ si } \texttt{DistL} \ \ge 25 \\ (511 - (D - 5) * 25\ ) \text{ sinon.} \end{cases} \tag{5.7}$$

La fonction **K** représente un coefficient d'atténuation de la luminosité en fonction de l'angle d'incidence de la source lumineuse :

$$\mathrm{K}(\texttt{ThetaL}, \theta_i) = 1 - \frac{1}{1 + e^{-4\beta(|\texttt{ThetaL} - \theta i| - \alpha)}} \qquad \alpha = 45 \text{ et } \beta = 0{,}03. \tag{5.8}$$

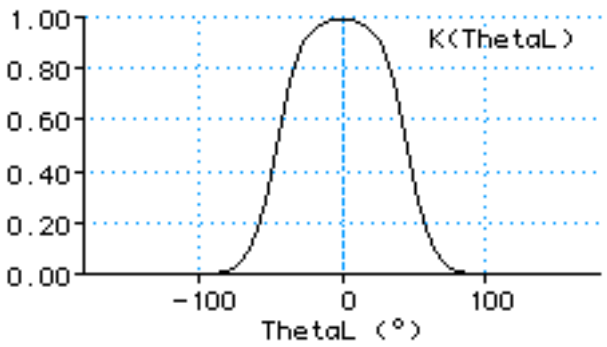


Figure 5-4 — Fonction **K**(ThetaL).
Cette fonction détermine le coefficient d'atténuation de la luminosité en fonction de l'angle d'incidence de la source lumineuse (ici, pour une position angulaire du capteur de luminosité $\theta_i = 0$).

Nous donnons donc ici explicitement et a priori les valeurs des paramètres des distributions utilisées dans les descriptions. Ces paramètres sont identiques pour les huit capteurs. Ils ont été déterminés à partir des courbes d'étalonnage fournies par le constructeur. Il serait cependant aisé d'identifier ces paramètres par l'expérience en calibrant indépendamment chacun des capteurs. Cette procédure permettrait notamment de prendre en compte les disparités de chaque capteur par rapport au modèle standard du constructeur.

Finalement, nous sommes en mesure de caractériser la réponse de n'importe quel capteur de luminosité *i* connaissant la position relative au robot d'une source lumineuse. Ainsi nous avons à notre disposition huit descriptions.

Par la suite chacune de ces huit descriptions sera identifiée par la connaissance préalable notée *Cp_li*. Ainsi la description du modèle de réponse du capteur *i* est résumée par :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL DistL Lmi} \mid Cp_li) \\ &= \mathrm{P}(\texttt{ThetaL} \mid Cp_li)\,\mathrm{P}(\texttt{DistL} \mid Cp_li)\,\mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ Cp_li). \end{aligned} \tag{5.9}$$

5.2.3 Utilisation

5.2.3.1 Programme et question probabiliste

Afin d'illustrer la nécessité de fusionner les informations provenant des capteurs, nous cherchons à calculer la connaissance sur ThetaL et sur DistL que nous pouvons déduire d'une description élémentaire *Cp_li*.

La question posée est P(ThetaL | Lmi *Cp_li*) afin de déterminer l'orientation de la source lumineuse.

Cette requête est décrit par :

1/ Description utilisée :

Cp_li

2/ Requête :

E_q : {ThetaL}

E_c : {Lmi}

E_i : {DistL}

Question : P(ThetaL | Lmi *Cp_li*)

3/ Décision :

Tirage aléatoire

La réponse à cette question est obtenue simplement par l'application de la règle du produit et la règle de marginalisation. Nous détaillons pour cette question les différentes étapes du calcul. La première étape consiste à appliquer la règle du produit :

$$\mathrm{P}(\mathtt{ThetaL} \mid \mathtt{Lmi}\ Cp_li) = \frac{\mathrm{P}(\mathtt{ThetaL}\ \mathtt{Lmi} \mid\ Cp_li)}{\mathrm{P}(\mathtt{Lmi} \mid\ Cp_li)}. \tag{5.10}$$

Puis, en appliquant la règle de marginalisation, nous pouvons exprimer le numérateur et le dénominateur en fonction de la distribution conjointe de la manière suivante :

$$\mathrm{P}(\mathtt{ThetaL} \mid \mathtt{Lmi}\ Cp_li) = \frac{\sum_{\mathtt{DistL}} \mathrm{P}(\mathtt{ThetaL}\ \mathtt{DistL}\ \mathtt{Lmi} \mid\ Cp_li)}{\sum_{\substack{\mathtt{DistL} \\ \mathtt{ThetaL}}} \mathrm{P}(\mathtt{ThetaL}\ \mathtt{DistL}\ \mathtt{Lmi} \mid\ Cp_li)}. \tag{5.11}$$

Nous remplaçons la distribution conjointe par sa décomposition définie dans les connaissances préalables :

$$\begin{aligned}&\mathrm{P}(\mathtt{ThetaL} \mid \mathtt{Lmi}\ Cp_li) \\ &= \frac{\sum_{\mathtt{DistL}} \mathrm{P}(\mathtt{DistL} \mid\ Cp_li)\mathrm{P}(\mathtt{ThetaL} \mid\ Cp_li)\mathrm{P}(\mathtt{Lmi} \mid\ \mathtt{ThetaL}\ \mathtt{DistL}\ Cp_li)}{\sum_{\substack{\mathtt{DistL} \\ \mathtt{ThetaL}}} \mathrm{P}(\mathtt{DistL} \mid\ Cp_li)\mathrm{P}(\mathtt{ThetaL} \mid\ Cp_li)\mathrm{P}(\mathtt{Lmi} \mid\ \mathtt{ThetaL}\ \mathtt{DistL}\ Cp_li)}.\end{aligned} \tag{5.12}$$

Une dernière simplification peut être appliquée sachant que les a priori sur DistL et ThetaL sont uniformes (P(ThetaL | *Cp_li*) = **U** et P(DistL | *Cp_li*) = **U**), et finalement la réponse à notre requête est :

$$P(\texttt{ThetaL} \mid \texttt{Lmi}\ Cp_li) = \frac{\sum_{\texttt{DistL}} P(\texttt{Lmi} \mid \texttt{ThetaL}\ \texttt{DistL}\ Cp_li)}{\sum_{\substack{\texttt{DistL} \\ \texttt{ThetaL}}} P(\texttt{Lmi} \mid \texttt{ThetaL}\ \texttt{DistL}\ Cp_li)}. \quad (5.13)$$

Afin de déterminer la distance de la source lumineuse, la question posée est P(DistL | Lmi *Cp_li*) :

1/ Description utilisée :

Cp_li

2/ Requête :

E_q : {DistL}

E_c : {Lmi}

E_i : {ThetaL}

Question : P(DistL | Lmi *Cp_li*)

3/ Décision :

Tirage aléatoire

De manière analogue au raisonnement suivi précédemment, la connaissance sur la distance de la source lumineuse P(DistL | Lmi *Cp_li*) est obtenue par l'expression suivante :

$$P(\texttt{DistL} \mid \texttt{Lmi}\ Cp_li) = \frac{\sum_{\texttt{ThetaL}} P(\texttt{Lmi} \mid \texttt{ThetaL}\ \texttt{DistL}\ Cp_li)}{\sum_{\substack{\texttt{ThetaL} \\ \texttt{DistL}}} P(\texttt{Lmi} \mid \texttt{ThetaL}\ \texttt{DistL}\ Cp_li)}. \quad (5.14)$$

5.2.3.2 Résultat

Les figures des pages suivantes (Figure 5-5 et Figure 5-6) présentent les distributions de probabilité sur l'orientation et la distance d'une source lumineuse déduite de la mesure d'un seul capteur.

Pour une luminosité forte (Lmi = 15), les réponses obtenues indiquent une source lumineuse face au capteur à une distance très proche, avec une incertitude relativement faible. Toutefois, au fur et à mesure que l'intensité diminue, l'incertitude augmente.

La distribution sur la distance semble toujours relativement uniforme entre une distance nulle et une borne maximum augmentant avec la valeurs lue sur le capteur de luminosité. Pour une luminosité très faible (Lmi ≥ 450) la connaissance sur la distance est très incertaine, la source lumineuse peut être à n'importe quelle distance.

La distribution de probabilité sur l'orientation fait apparaître deux pics symétriques s'écartant du centre avec la baisse de luminosité. La lecture d'un seul capteur ne permet pas de décider de quel coté provient la lumière. La probabilité d'observer la source lumineuse face au robot décroît pour s'annuler complètement pour des valeurs de Lmi ≥ 475. Au delà, la

réponse fournit une information proscriptive, la source peut être n'importe où hormis dans un secteur face au robot.

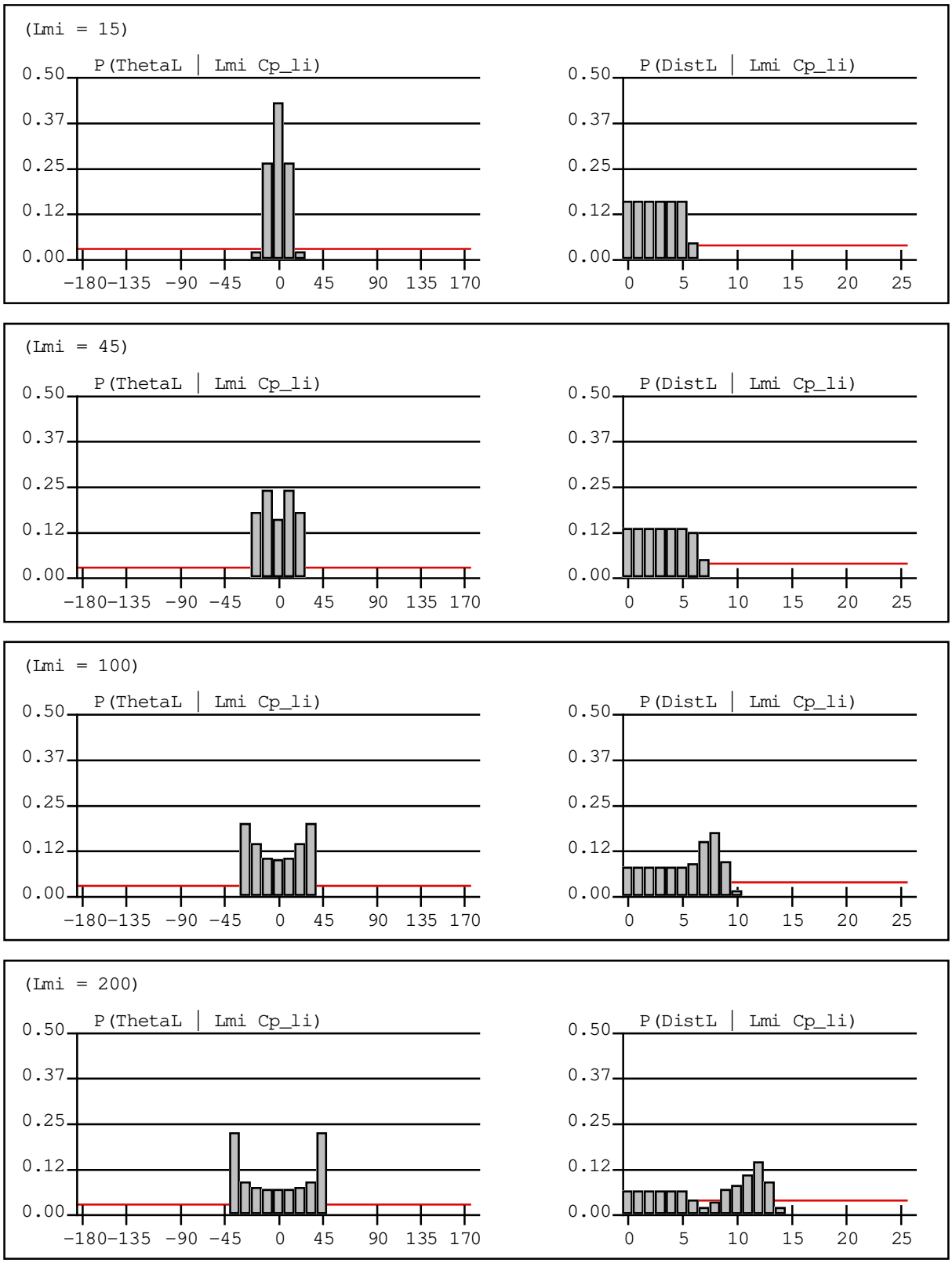


Figure 5-5 — Connaissance sur l'orientation et la distance d'une source lumineuse.
Cette connaissance est déduite de la mesure d'un seul capteur pour différents valeurs lues (15, 45, 100, 200) correspondant à une luminosité importante.

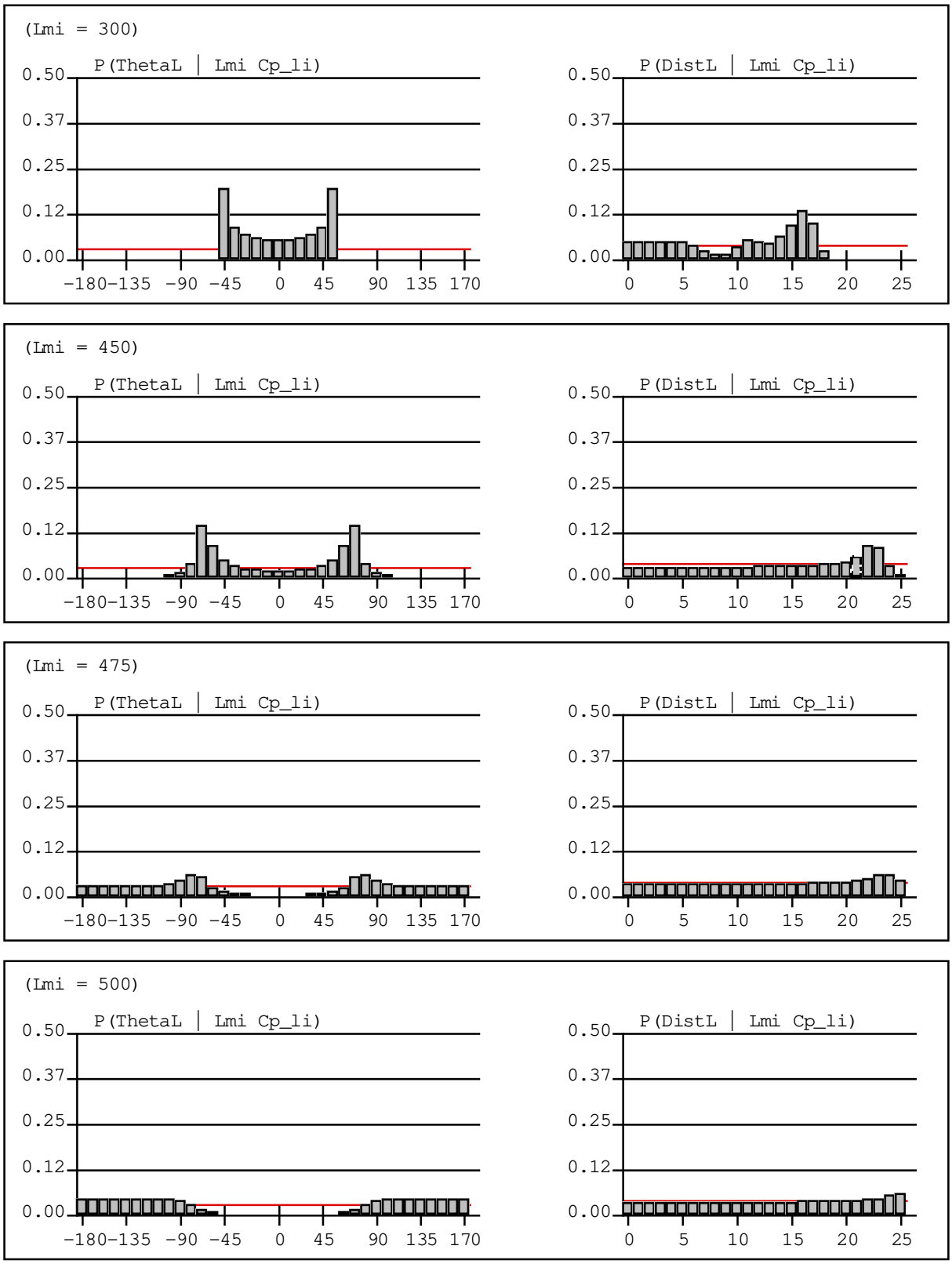


Figure 5-6 — Connaissance sur l'orientation et la distance d'une source lumineuse.
Cette connaissance est déduite de la mesure d'un seul capteur pour différents valeurs lues (300, 450, 475, 500) correspondant à une luminosité faible.

5.3 Phase 2 : fusion des capteurs

5.3.1 Spécification

5.3.1.1 Choix des variables

Les variables retenues sont les deux variables caractérisant la source lumineuse `ThetaL`, `DistL`, ainsi que les 8 variables correspondant aux huit capteurs de luminosité :`Lm0` à `Lm7`.

Variables pertinentes :

- **`ThetaL`** : domaine de valeur {-180, -170,…, +170}, $k_{\texttt{ThetaL}} = 36$ cas ;
- **`DistL`** : domaine de valeur {0, 1,…, 25}, $k_{\texttt{DistL}} = 26$ cas ;
- **`Lm0 à Lm7`** : domaine de valeur {0, 1,…, 511}, $k_{\texttt{Lmi}} = 512$ cas.

5.3.1.2 Connaissances préalables de dépendance

La description du modèle porte sur les variables `ThetaL`, `DistL` et `Lm0` à `Lm7` et est associée à la distribution conjointe suivante :

$$\mathrm{P}(\texttt{ThetaL DistL Lm0 Lm1 Lm2 Lm3 Lm4 Lm5 Lm6 Lm7} \mid D\ C). \tag{5.15}$$

Dans un premier temps, nous décomposons la distribution conjointe en appliquant la règle **[R2]** :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid D\ C) \\ &= \begin{vmatrix} \mathrm{P}(\texttt{ThetaL} \mid D\ C)\mathrm{P}(\texttt{DistL} \mid \texttt{ThetaL}\ D\ C) \\ \mathrm{P}(\texttt{Lm0..Lm7} \mid \texttt{ThetaL DistL}\ D\ C). \end{vmatrix} \end{aligned} \tag{5.16}$$

Nous considérons à nouveau que la connaissance de la direction de la source lumineuse n'apporte aucune information sur la distance à laquelle elle se trouve par rapport au robot. Nous traduisons cette a priori en considérant que la probabilité de `DistL` est indépendante de `ThetaL` :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid D\ C) \\ &= \mathrm{P}(\texttt{ThetaL} \mid D\ C)\mathrm{P}(\texttt{DistL} \mid D\ C)\mathrm{P}(\texttt{Lm0..Lm7} \mid \texttt{ThetaL DistL}\ D\ C). \end{aligned} \tag{5.17}$$

Il reste à décomposer la distribution conjointe portant sur les huit variables `Lm0` à `Lm7`. Pour cela, nous allons procéder en deux temps. Premièrement, nous décomposons cette expression en un produit de huit distributions élémentaires en appliquant la règle **[R2]** :

$$
\begin{aligned}
&\mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid D\ C) \\
&= \left| \begin{array}{l}
\mathrm{P}(\texttt{ThetaL} \mid D\ C)\mathrm{P}(\texttt{DistL} \mid D\ C) \\
\mathrm{P}(\texttt{Lm0} \mid \texttt{ThetaL DistL}\ D\ C) \\
\mathrm{P}(\texttt{Lm1} \mid \texttt{Lm0 ThetaL DistL}\ D\ C) \\
\mathrm{P}(\texttt{Lm2} \mid \texttt{Lm0 Lm1 ThetaL DistL}\ D\ C) \\
\quad ... \\
\mathrm{P}(\texttt{Lm7} \mid \texttt{Lm0..Lm6 ThetaL DistL}\ D\ C).
\end{array} \right.
\end{aligned}
\tag{5.18}
$$

Finalement, nous allons exprimer des connaissances sous la forme d'hypothèses d'indépendance conditionnelle. Il apparaît clairement que les mesures de luminosité lues sur deux capteurs contigus ne sont pas indépendantes l'une de l'autre. En effet, si un capteur de luminosité est fortement éclairé, il est probable qu'un capteur à proximité de celui-ci le soit aussi. Toutefois, si nous sommes en mesure de caractériser avec certitude la position de la source lumineuse, alors nous pouvons supposer que la dépendance entre deux capteurs de luminosité est principalement résumée par la connaissance de la source. En d'autres termes, la dépendance qui peut exister entre les valeurs de deux capteurs provient du fait que ces valeurs sont le résultat d'un même phénomène ; ici, la source lumineuse. Ainsi, conditionnellement à la connaissance caractérisant la source lumineuse, nous sommes en mesure de considérer les valeurs des capteurs indépendantes les unes des autres. Ce qui se traduit sous la forme suivante :

$$
\begin{aligned}
&\mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid D\ C) \\
&= \left| \begin{array}{l}
\mathrm{P}(\texttt{ThetaL} \mid D\ C)\mathrm{P}(\texttt{DistL} \mid D\ C) \\
\mathrm{P}(\texttt{Lm0} \mid \texttt{ThetaL DistL}\ D\ C) \\
\mathrm{P}(\texttt{Lm1} \mid \texttt{ThetaL DistL}\ D\ C) \\
\mathrm{P}(\texttt{Lm2} \mid \texttt{ThetaL DistL}\ D\ C) \\
\quad ... \\
\mathrm{P}(\texttt{Lm7} \mid \texttt{ThetaL DistL}\ D\ C).
\end{array} \right.
\end{aligned}
\tag{5.19}
$$

Cette décomposition est intéressante dans la mesure où elle permet d'exprimer d'une part nos a priori sur la distance et la direction de la source lumineuse par rapport au robot et, d'autre part, le produit des distributions décrivant la réponse `Lmi` de chacun des capteurs de luminosité connaissant les informations relatives à la source lumineuse. Ce produit peut être vu comme une manière de fusionner les différentes informations provenant des capteurs de luminosité.

5.3.1.3 Connaissances préalables d'observation

Ici nous n'allons pas identifier expérimentalement les différentes distributions élémentaires formant la décomposition. En effet, nous considérons que cette phase a déjà été réalisée lors de l'identification des modèles de la réponse de chacun des capteurs et est résumée dans les connaissances préalables *Cp_li*. Ainsi, il nous reste juste à exprimer que les distributions caractérisant la réponse des capteurs de luminosité sont obtenues en posant la question P(Lmi | ThetaL DistL *Cp_li*) à chaque description préalablement donnée *Cp_li* :

$$
\begin{aligned}
\mathrm{P}(\texttt{ThetaL} \mid D\ C) &= \mathbf{U}_{k_{\texttt{ThetaL}}} = \frac{1}{36}, \\
\mathrm{P}(\texttt{DistL} \mid D\ C) &= \mathbf{U}_{k_{\texttt{DistL}}} = \frac{1}{26}, \\
\mathrm{P}(\texttt{Lm0} \mid \texttt{ThetaL DistL}\ D\ C) &= \mathrm{P}(\texttt{Lm0} \mid \texttt{ThetaL DistL}\ Cp_l0), \\
\mathrm{P}(\texttt{Lm1} \mid \texttt{ThetaL DistL}\ D\ C) &= \mathrm{P}(\texttt{Lm1} \mid \texttt{ThetaL DistL}\ Cp_l1), \\
\mathrm{P}(\texttt{Lm2} \mid \texttt{ThetaL DistL}\ D\ C) &= \mathrm{P}(\texttt{Lm2} \mid \texttt{ThetaL DistL}\ Cp_l2), \\
&\ldots \\
\mathrm{P}(\texttt{Lm7} \mid \texttt{ThetaL DistL}\ D\ C) &= \mathrm{P}(\texttt{Lm7} \mid \texttt{ThetaL DistL}\ Cp_l7).
\end{aligned}
\tag{5.20}
$$

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_SL* :

$$
\mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid Cp_SL) = \begin{vmatrix}
\mathrm{P}(\texttt{ThetaL} \mid Cp_SL)\,\mathrm{P}(\texttt{DistL} \mid Cp_SL) \\
\mathrm{P}(\texttt{Lm0} \mid \texttt{ThetaL DistL}\ Cp_SL) \\
\mathrm{P}(\texttt{Lm1} \mid \texttt{ThetaL DistL}\ Cp_SL) \\
\mathrm{P}(\texttt{Lm2} \mid \texttt{ThetaL DistL}\ Cp_SL) \\
\ldots \\
\mathrm{P}(\texttt{Lm7} \mid \texttt{ThetaL DistL}\ Cp_SL).
\end{vmatrix}
\tag{5.21}
$$

5.3.2 Utilisation

5.3.2.1 Programme et question probabiliste

La question que l'on veut se poser est P(`ThetaL` | `Lm0…Lm7` *Cp_SL*) afin de déterminer l'orientation de la source lumineuse relativement au robot. Cette requête à la description *Cp_SL* nécessite implicitement l'utilisation des 8 descriptions *Cp_l0* à *Cp_l7*.

1/ Description utilisée :

Cp_SL, Cp_l0, Cp_l1, …Cp_l7

2/ Requête :

E_q : {`ThetaL`}

E_c : {`Lm0`, `Lm1`, `Lm2`, … `Lm7`}

E_i : {`DistL`}

Question : P(`ThetaL` | `Lm0 Lm1 Lm2 …Lm7` *Cp_SL*)

3/ Décision :

Tirage aléatoire

La réponse à cette question est obtenue simplement par l'application de la règle du produit et la règle de marginalisation. À nouveau la première étape consiste à appliquer la règle du produit :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm0..Lm7}\ \mathit{Cp_SL}) \\ &= \frac{\mathrm{P}(\texttt{ThetaL Lm0..Lm7} \mid \mathit{Cp_SL})}{\mathrm{P}(\texttt{Lm0..Lm7} \mid \mathit{Cp_SL})}. \end{aligned} \tag{5.22}$$

Puis en appliquant la règle de marginalisation, nous pouvons exprimer le numérateur et le dénominateur en fonction de la distribution conjointe de la manière suivante :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm0..Lm7}\ \mathit{Cp_SL}) \\ &= \frac{\sum_{\texttt{DistL}} \mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid \mathit{Cp_SL})}{\sum_{\substack{\texttt{DistL} \\ \texttt{ThetaL}}} \mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid \mathit{Cp_SL})}. \end{aligned} \tag{5.23}$$

Nous remplaçons la distribution conjointe par sa décomposition définie dans les connaissances préalables et nous simplifions l'expression sachant que les a priori sur `DistL` et `ThetaL` sont uniformes ($\mathrm{P}(\texttt{ThetaL} \mid \mathit{Cp_li}) = \mathbf{U}$ et $\mathrm{P}(\texttt{DistL} \mid \mathit{Cp_li}) = \mathbf{U}$) et donc peuvent être mis en facteur des sommes. Finalement la réponse à notre requête est :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm0..Lm7}\ \mathit{Cp_SL}) \\ &= \frac{\sum_{\texttt{DistL}} \prod_{i=0}^{7} \mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ \mathit{Cp_li})}{\sum_{\substack{\texttt{DistL} \\ \texttt{ThetaL}}} \prod_{i=0}^{7} \mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ \mathit{Cp_li})}. \end{aligned} \tag{5.24}$$

Le dénominateur de cette expression représente une constante de normalisation ne dépendant que des valeurs particulières des capteurs `Lm0`…`Lm7`. En notant Z cette constante, la réponse à notre question devient :

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm0..Lm7}\ \mathit{Cp_SL}) \\ &= \frac{1}{Z} \sum_{\texttt{DistL}} \prod_{i=0}^{7} \mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ \mathit{Cp_li}). \end{aligned} \tag{5.25}$$

Voir le résultat à la section 5.3.2.2.

De nombreuses autres requêtes sont envisageables.

Par exemple nous pouvons chercher l'orientation de la source lumineuse par rapport au robot ne connaissant que les valeurs de certains des capteurs : $\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm1 Lm2}\ \mathit{Cp_SL})$.

1/ Description utilisée :

Cp_SL, Cp_l0, Cp_l1, …Cp_l7

2/ Requête :

E_q : {ThetaL}

E_c : {Lm1, Lm2}

E_i : {DistL, Lm0, Lm3, …Lm7}

Question : P(ThetaL | Lm1 Lm2 *Cp_SL*)

3/ Décision :

Tirage aléatoire

La réponse est alors obtenue par :

$$\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm1 Lm2}\ Cp_SL) = \frac{\mathrm{P}(\texttt{ThetaL Lm1 Lm2} \mid Cp_SL)}{\mathrm{P}(\texttt{Lm1 Lm2} \mid Cp_SL)}. \tag{5.26}$$

Qui, en suivant un résonnement similaire que précédemment, se simplifie en :

$$\begin{aligned}&\mathrm{P}(\texttt{ThetaL} \mid \texttt{Lm1 Lm2}\ Cp_SL)\\ &= \frac{1}{Z}\sum_{\texttt{DistL}} \mathrm{P}(\texttt{Lm1} \mid \texttt{ThetaL DistL}\ Cp_l1)\,\mathrm{P}(\texttt{Lm2} \mid \texttt{ThetaL DistL}\ Cp_l2).\end{aligned} \tag{5.27}$$

Nous remarquons dans cette expression que, finalement, se questionner sur la source en ne prenant en compte qu'une partie des informations sensorielles (Lm1 et Lm2) revient implicitement et automatiquement à ne fusionner que les connaissances relatives aux descriptions associées aux capteurs retenus (*Cp_l1* et *Cp_l2*).

Les figures suivantes présentent les réponses pour deux situations différentes.

La Figure 5-7 correspond à une situation où la source est située 90° à droite du robot. La réponse obtenue par fusion sur les capteurs 1 et 2 ne permet pas de positionner avec certitude la source ; par contre, cette fusion accroit la zone dans laquelle on sait avec certitude que la source n'est pas.

La Figure 5-8 correspond à une situation où la source est située à gauche du robot (-50°). Cette fois ci, la source se trouve dans un secteur mieux adapté aux capteurs retenus. La réponse obtenue par fusion sur les capteurs 1 et 2 est beaucoup plus prononcée et indique la bonne orientation avec une plausibilité relativement importante.

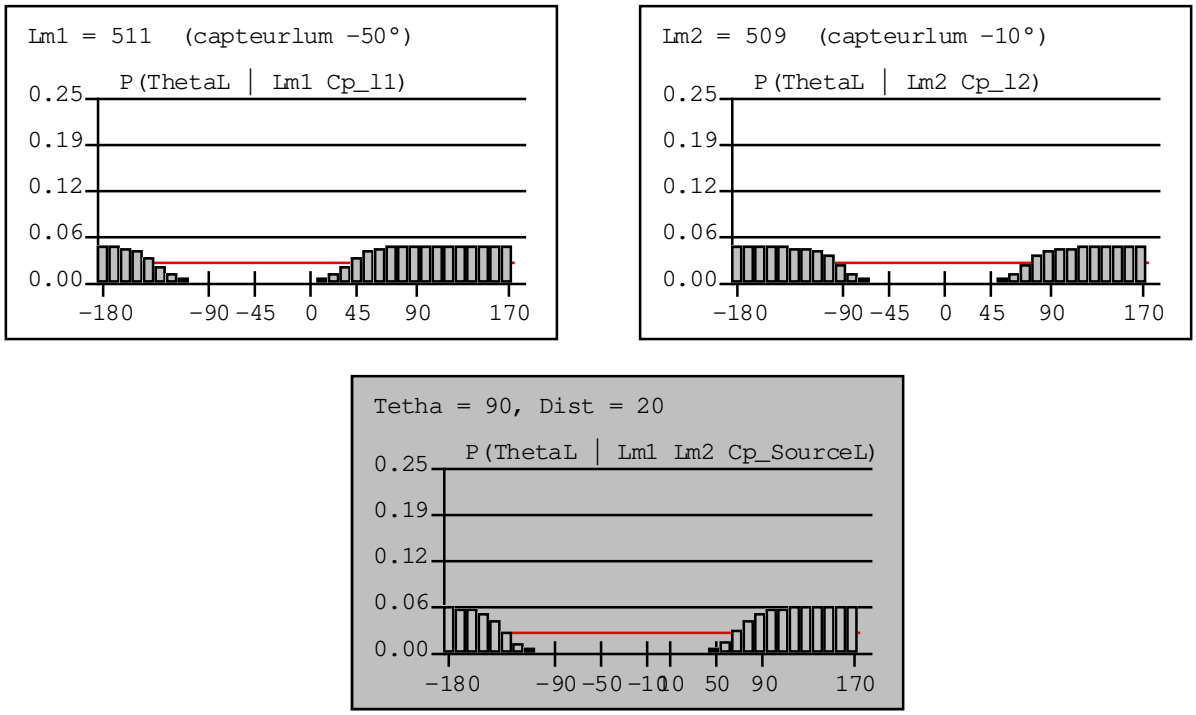


Figure 5-7 — Connaissance sur l'orientation de la source lumineuse.
Cette connaissance est obtenue en ne prenant en compte que les informations fournies par les capteurs 1 et 2 (les paramètres de la source lumineuse étant fixés à `ThetaL` = 90° et `DistL` = 20)

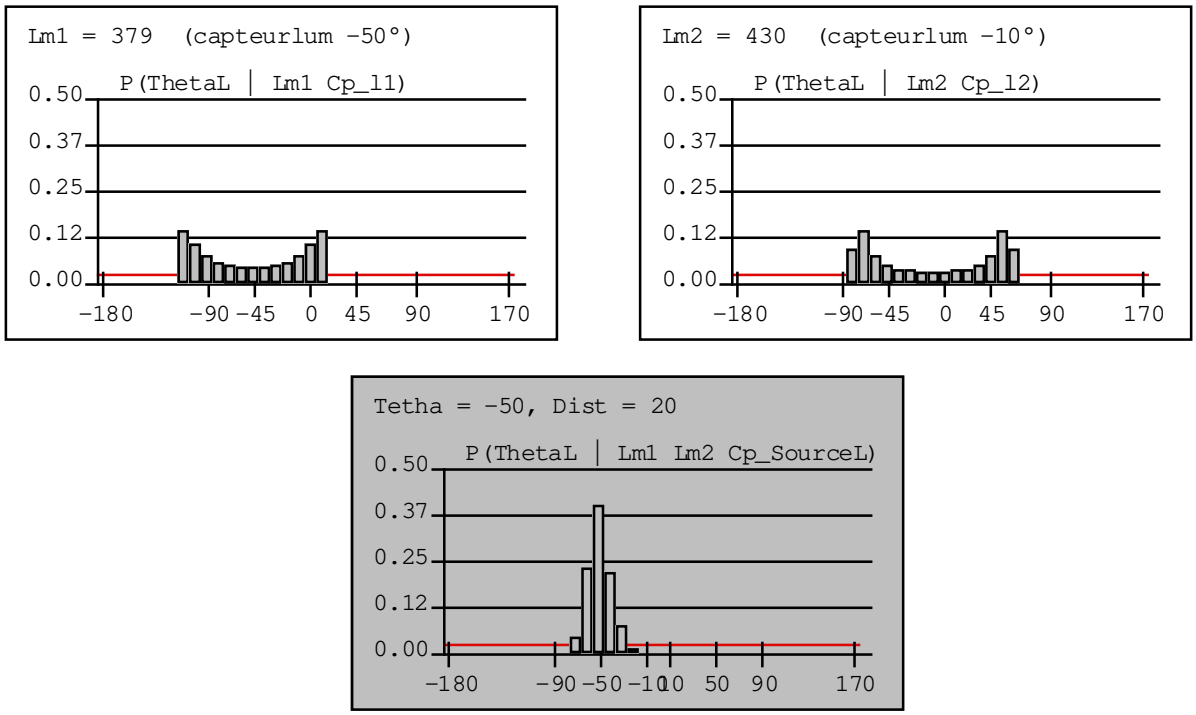


Figure 5-8 — Connaissance sur l'orientation de la source lumineuse.
Cette connaissance est obtenue en ne prenant en compte que les informations fournies par les capteurs 1 et 2 (les paramètres de la source lumineuse étant fixés à `ThetaL` = -50° et `DistL` = 20)[3]

[3] Le résultat présenté peut surprendre à première vue, en particulier la fusion fixe une probabilité nulle pour [Theta=0] bien que ni (P(`Theta` | `Lm1` *Cp_l1*) ni P(`Theta` | `Lm2` *Cp_l2*)) ne soit nulle pour ce cas. Il faut bien prendre ne garde que la fusion est obtenue par la relation (5.27) qui est différente du produit P(`ThetahL` | `Lm1` *CpL1*)× P(`ThetahL` | `Lm2` *CpL2*).

Nous pouvons aussi nous poser des questions sur la distance de la source connaissant son orientation et les lectures capteurs : P(DistL | ThetaL Lm0…Lm7 *Cp_SL*) :

1/ Descriptions utilisées :

Cp_SL, Cp_l0, Cp_l1, …Cp_l7

2/ Requête :

E_q : {DistL}

E_c : {ThetaL, Lm0, Lm1, Lm2, … Lm7}

E_i : Ø

Question : P(DistL | ThetaL Lm0 … Lm7 *Cp_SL*)

3/ Décision :

Tirage aléatoire

La réponse est obtenue par :

$$P(\texttt{DistL} \mid \texttt{ThetaL Lm0..Lm7}\ Cp_SL) = \frac{P(\texttt{ThetaL DistL Lm0..Lm7} \mid\ Cp_SL)}{P(\texttt{ThetaL Lm0..Lm7} \mid\ Cp_SL)}. \quad (5.28)$$

Et, après simplification, s'exprime ainsi :

$$P(\texttt{DistL} \mid \texttt{ThetaL Lm0..Lm7}\ Cp_SL) = \frac{1}{Z}\prod_{i=0}^{7} P(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ Cp_li). \quad (5.29)$$

Nous pouvons, enfin, nous poser des questions « inverses » visant à « prédire » les lectures capteurs connaissant, par exemple, l'orientation de la source et les lectures d'autres capteurs : P(Lm3 | ThetaL Lm2 Lm4 *Cp_SL*).

1/ Description utilisée :

Cp_SL, Cp_l0, Cp_l1, …Cp_l7

2/ Requête :

E_q : {Lm3 }

E_c : {ThetaL, Lm2, Lm4}

E_i : {DistL, Lm0, Lm1, Lm5, Lm6, Lm7}

Question : P(Lm3 | ThetaL Lm2 Lm4 *Cp_SL*)

3/ Décision :

Tirage aléatoire

La réponse est obtenue par :

$$\mathrm{P}(\texttt{Lm3} \mid \texttt{ThetaL Lm2 Lm4}\ Cp_SL) = \frac{\mathrm{P}(\texttt{ThetaL Lm2 Lm3 Lm4} \mid Cp_SL)}{\mathrm{P}(\texttt{ThetaL Lm2 Lm4} \mid Cp_SL)}. \tag{5.30}$$

L'expression se simplifie en :

$$\begin{aligned} &\mathrm{P}(\texttt{Lm3} \mid \texttt{ThetaL Lm2 Lm4}\ Cp_SL) \\ &= \frac{1}{Z} \sum_{\texttt{DistL}} \prod_{i=2}^{4} \mathrm{P}(\texttt{Lmi} \mid \texttt{ThetaL DistL}\ Cp_li). \end{aligned} \tag{5.31}$$

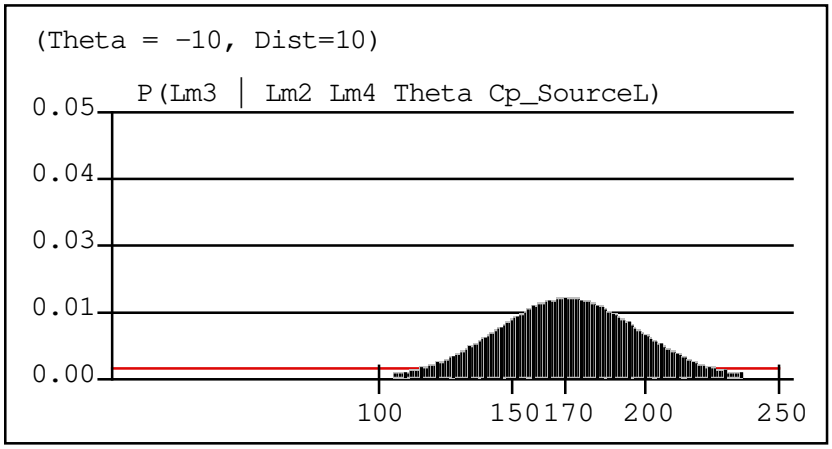


Figure 5-9 — Réponse prédite du capteur `Lm3`.
Cette réponse est calculée connaissant les valeurs fournies par les capteur 2 et 4 ainsi que l'orientation de la source lumineuse, la distance étant ignorée. Nous observons que la probabilité maximale correspond à la réponse théorique pour une distance `Dist` = 10 : `Lum3` = 170

5.3.2.2 Résultat

Cette méthode de fusion de capteurs est extrêmement efficace. Ses avantages sont multiples :

- Elle permet de rehausser le signal très efficacement.
- Elle est robuste aux bruits ou à la défaillance de certains capteurs.
- Elle permet, à partir de plusieurs capteurs de piètre qualité, d'obtenir par fusion une information plus précise.

Les deux figures suivantes montrent les réponses des huit capteurs de luminosité et le résultat de leur fusion pour une source lumineuse situé à 10° sur la droite du Khepera, à une distance de 20 (Figure 5-10) et une distance de 10 (Figure 5-11).

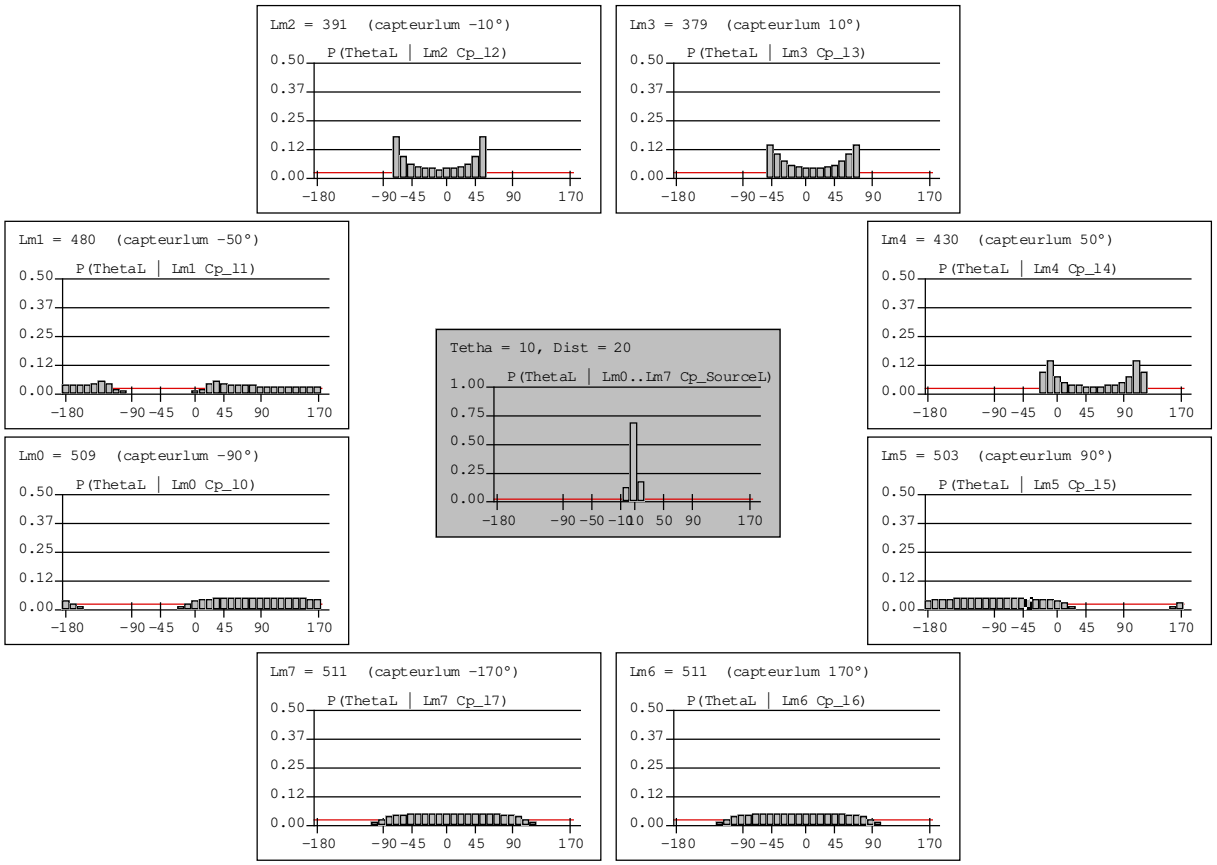


Figure 5-10 — Fusion des huit capteurs pour `Theta` = 10 et `DistL` = 20.

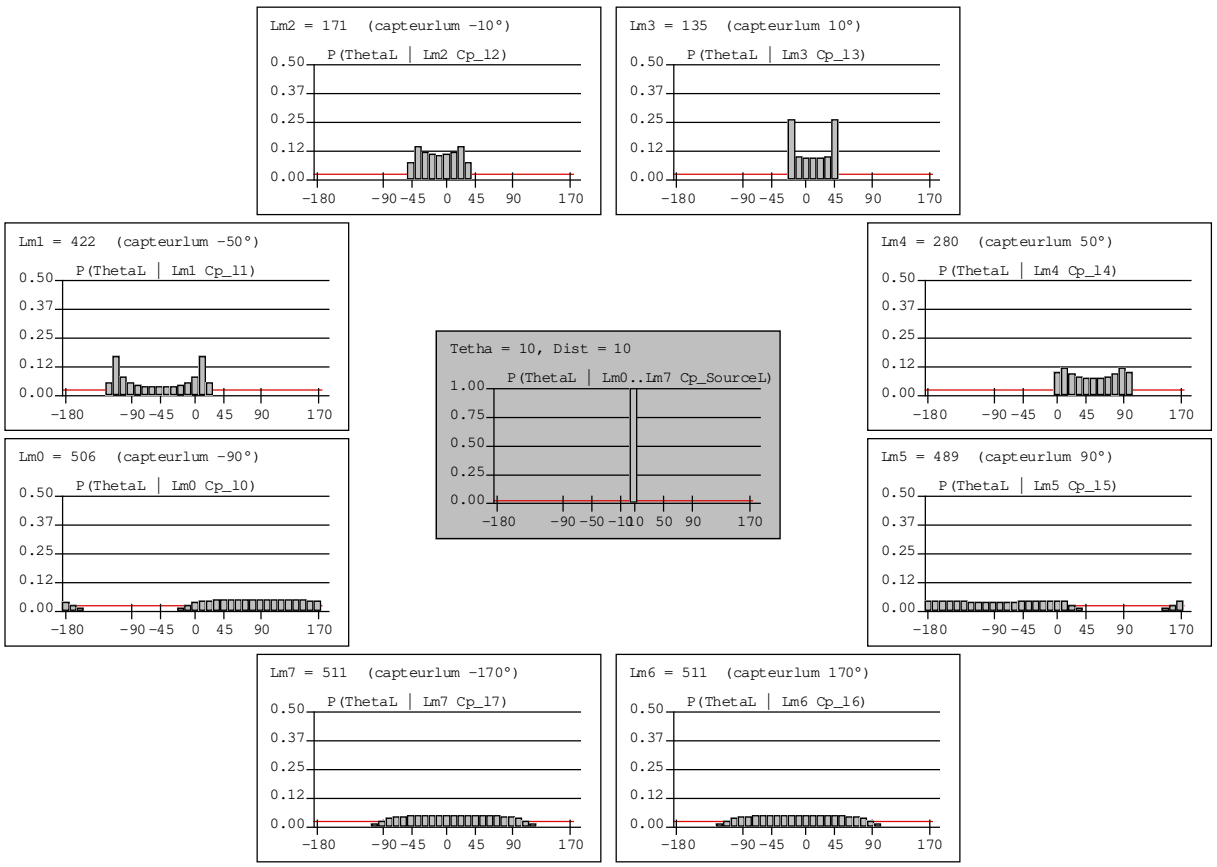


Figure 5-11 — Fusion des huit capteurs pour `Theta` = 10 et `DistL` = 10.

L'information fournie par chacun des huit capteurs pris isolément est très médiocre. Les deux capteurs situés à l'arrière et les 2 capteurs latéraux, peu éclairés, correspondent à des distributions de probabilités indiquant seulement qu'il y a très peu de chance pour que la source lumineuse se trouve en face d'eux. Les deux capteurs situés à –45° et +45°, faiblement éclairés, correspondent à des distributions qui montrent que la source de lumière n'est pas en face d'eux mais qui n'excluent pas qu'elle puisse se trouver en lisière de leurs champs perceptifs.

Le résultat (présenté au centre de la figure) de la fusion probabiliste de ces huit capteurs est saisissant ! Il s'agit d'un Dirac sur la position +10°. La position de la source est donc non seulement trouvée avec exactitude, mais, aussi, avec certitude.

5.4 Enseignement et discussion

De nombreux enseignements peuvent être tirés de cette expérience.

Tout d'abord, la fusion de capteurs, faite de cette manière, entraîne d'une part un rehaussement très important du signal lorsque les capteurs sont en accord et, d'autre part, une résistance importante aux « bruits » perturbant certains des capteurs.

Ensuite, l'approche probabiliste de la fusion des informations capteurs présente l'avantage de pouvoir répondre à n'importe quel type de question concernant le modèle. Comme nous l'avons vu, il est possible de déterminer `ThetaL` en fonction des lectures capteurs mais aussi, de déterminer `ThetaL` ne connaissant qu'une partie de ces lectures ou encore de prédire les lectures capteurs sachant `ThetaL`. Nous ne sommes pas contraint par un modèle direct ne permettant de répondre qu'à certaines de ces questions. Dans la distribution conjointe : P(`DistL ThetaL Lm0…Lm7` | *Cp_SL*) toutes les variables jouent mathématiquement le même rôle et les notions même de modèle direct ou inverse n'ont plus vraiment de sens.

Bien que l'on puisse théoriquement répondre à toutes les questions, il faut bien voir qu'en pratique, il peut être plus difficile de répondre à certaines questions qu'à d'autres. En particulier, comme nous l'avons vu, certaines questions supposent d'effectuer des sommes aves un grand nombre de termes. Il faut en effet sommer sur tous les cas possibles de toutes les variables inconnues. Le calcul estimé de ces sommes est l'une des difficultés principales que le moteur d'inférence doit résoudre. Ce problème est en fait, essentiellement, un problème d'optimisation que le moteur aborde notamment à l'aide d'algorithmes génétiques.

Une condition *sine qua non* de cette approche de la fusion de capteurs est d'apporter d'importantes simplifications à la décomposition de la distribution conjointe. Ces simplifications, rappelons le, viennent d'hypothèses d'indépendance conditionnelle entre variables. Dans cette expérience, nous avons fait l'hypothèse que les capteurs de luminosité ne dépendent les uns des autres uniquement par la relation qu'ils entretiennent avec une unique source de lumière. Cette hypothèse permet de passer d'une distribution sur un espace de dimension 10 et de taille $36x26x512^8$ à un produit de 9 distributions beaucoup plus simples sur des espaces de dimension 3 et de taille 36x26x512. Elle est générale car ce type d'hypothèse peut être fait pour la plupart des ensembles de capteurs répondant à un même phénomène.

Enfin, un autre point à noter dans cette expérience est la manière dont ont été pris en compte les modèles des capteurs. En effet, lorsque l'on décompose la distribution conjointe comme expliqué dans la section précédente, on voit apparaître les distributions P(`Lmi` | `DistL ThetaL`) qui correspondent précisément à un modèle probabiliste des

capteurs. Nous avons vu que ces modèles sont faciles à donner a priori car ils correspondent, très souvent, à des données fournies par le constructeur des dits capteurs. Nous avons vu aussi qu'il est facile, si cela s'avère nécessaire, de calibrer individuellement ces capteurs en identifiant les paramètres des distributions P(`Lmi` | `DistL ThetaL`) sur la base de jeux d'expériences propres à chacun d'eux.

Chapitre 6

Combinaison de comportements

Un des enjeux d'un système de programmation de robot est d'être en mesure de construire des comportements plus complexes en combinant des ressources comportementales préalablement acquises. Nous présentons ici deux expériences illustrant comment aborder cette idée dans le cadre probabiliste.

6.1 Tourner et ralentir en présence d'obstacles

6.1.1 Objectifs et protocole expérimental

L'objectif de cette première expérience est d'obtenir du Khepera, lorsqu'il est en évitement d'obstacle, qu'il module sa vitesse en fonction de la proximité des obstacles. Nous désirons qu'il se déplace rapidement en espace libre et qu'il ralentisse lorsqu'il est plus près des obstacles.

Nous voulons obtenir ce résultat sans nouvel apprentissage, en combinant entre eux des comportements déjà existants, d'une part l'évitement d'obstacle à vitesse constante (*Cp_Tourner*) décrit au Chapitre 4 section 4.3.1 et d'autre part la modulation de la vitesse de translation en présence d'obstacle (*Cp_Ralentir*) décrite au Chapitre 4 section 4.3.2.

Nous allons voir que cela peut être réalisé grâce à une combinaison probabiliste de ces deux descriptions.

6.1.2 Spécification

6.1.2.1 Choix des variables

Les variables retenues sont l'union des variables des deux descriptions déjà existantes, c'est-à-dire : `Vrot`, `Vtrans`, `Vtrans_c`, `Dir` et `Prox`.

6.1.2.2 Connaissances préalables de dépendance

La distribution conjointe correspondant à la nouvelle description est :

$$\mathrm{P}\left(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid D\ C\right). \tag{6.1}$$

Nous décidons de décomposer cette distribution de la manière suivante :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid D\ C\right)\\ &= \begin{vmatrix}\mathrm{P}\left(\texttt{Dir Prox Vrot} \mid D\ C\right)\\ \mathrm{P}\left(\texttt{Vtrans_c Vtrans} \mid \texttt{Dir Prox Vrot}\ D\ C\right).\end{vmatrix}\end{aligned} \tag{6.2}$$

En posant l'hypothèse que, connaissant `Prox`, `Vtrans` et `Vtrans_c` ne dépendent pas de `Vrot` et `Dir`, cette décomposition se simplifie en :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid D\ C\right)\\ &= \begin{vmatrix}\mathrm{P}\left(\texttt{Dir Prox Vrot} \mid D\ C\right)\\ \mathrm{P}\left(\texttt{Vtrans_c Vtrans} \mid \texttt{Prox}\ D\ C\right).\end{vmatrix}\end{aligned} \tag{6.3}$$

Cette décomposition se résume au produit de deux distributions élémentaires que l'on peut aisément associer aux descriptions *Cp_Tourner* et *Cp_Ralentir*.

6.1.2.3 Connaissances préalables d'observation

Ce sont les connaissances des deux descriptions utilisées.

$$\mathrm{P}\left(\texttt{Dir Prox Vrot} \mid D\ C\right) = \mathrm{P}\left(\texttt{Dir Prox Vrot} \mid Cp_Tourner\right), \tag{6.4}$$

$$\mathrm{P}\left(\texttt{Vtrans_c Vtrans} \mid \texttt{Prox}\ D\ C\right) = \mathrm{P}\left(\texttt{Vtrans_c Vtrans} \mid \texttt{Prox}\ Cp_Ralentir\right). \tag{6.5}$$

La première distribution élémentaire correspond à la description *Cp_Tourner*, et la deuxième à une question posée à la description *Cp_Ralentir*, consistant à chercher `Vtrans` et `Vtrans_c` connaissant la valeur de la variable `Prox`.

6.1.3 Identification

Pas de nouvelle identification, nous utilisons celles qui ont déjà été faites pour les deux descriptions utilisées.

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_Evit* :

$$\mathrm{P}\left(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid \mathit{Cp_Evit}\right) = \begin{vmatrix} \mathrm{P}\left(\texttt{Dir Prox Vrot} \mid \mathit{Cp_Tourner}\right) \\ \mathrm{P}\left(\texttt{Vtrans_c Vtrans} \mid \texttt{Prox } \mathit{Cp_Ralentir}\right). \end{vmatrix} \quad (6.6)$$

6.1.4 Utilisation

6.1.4.1 Programme et question probabiliste

La question probabiliste pour commander le robot avec cette nouvelle description consiste à déterminer les vitesses de rotation et de translation connaissant les valeurs des autres variables `Dir`, `Prox` et `Vtrans_c`. Cette requête est décrite par :

1/ Descriptions utilisées :

Cp_Evit, Cp_Tourner, Cp_ralentir

2/ Requête :

E_q : {Vtrans, Vrot}

E_c : {Dir, Prox, Vtrans_c}

E_i : Ø

Question : P(Vtrans Vrot | Dir Prox Vtrans_c *Cp_Evit*)

3/ Décision :

Tirage aléatoire

Nous allons détailler la résolution de la question. Par la suite, nous noterons les variables dont les valeurs sont connues (ici `Dir`, `Prox` et `Vtrans_c`) en minuscule (`dir`, `prox` et `vtrans_c`) afin de rappeler que ces variables ont une valeur particulière lors de la question :

$$\texttt{dir} \equiv [\texttt{Dir} = \texttt{dir}_i] \text{ avec } \texttt{dir}_i \in D_{\texttt{Dir}}$$

Cette question se résout en appliquant la règle du produit de la manière suivante :

$$\mathrm{P}\left(\texttt{Vtrans Vrot} \mid \texttt{dir prox vtrans_c } \mathit{Cp_Evit}\right) = \frac{\mathrm{P}\left(\texttt{dir prox vtrans_c Vtrans Vrot} \mid \mathit{Cp_Evit}\right)}{\mathrm{P}\left(\texttt{dir prox vtrans_c} \mid \mathit{Cp_Evit}\right)}. \quad (6.7)$$

Le dénominateur s'obtient par la marginalisation de la distribution conjointe sur les variables `Vrot` et `Vtrans` :

$$\begin{aligned}&\mathrm{P}\big(\texttt{Vtrans Vrot} \mid \texttt{dir prox vtrans_c } \mathit{Cp_Evit}\big)\\ &= \frac{\mathrm{P}\big(\texttt{dir prox vtrans_c Vtrans Vrot} \mid \mathit{Cp_Evit}\big)}{\sum_{\substack{\texttt{Vtrans}\\ \texttt{Vrot}}} \mathrm{P}\big(\texttt{dir prox vtrans_c Vtrans Vrot} \mid \mathit{Cp_Evit}\big)}.\end{aligned} \tag{6.8}$$

Puis nous remplaçons la distribution conjointe par son expression définie en (6.6) :

$$\begin{aligned}&\mathrm{P}\big(\texttt{Vtrans Vrot} \mid \texttt{dir prox vtrans_c } \mathit{Cp_Evit}\big)\\ &= \frac{\mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)}{\sum_{\substack{\texttt{Vtrans}\\ \texttt{Vrot}}} \mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)}.\end{aligned} \tag{6.9}$$

Le dénominateur peut être simplifié. L'expression P(`Vrot dir prox` | *Cp_Evit*) ne dépend pas de la variable `Vtrans`, elle peut donc être mise en facteur de la marginalisation sur `Vtrans` :

$$\begin{aligned}&\sum_{\substack{\texttt{Vtrans}\\ \texttt{Vrot}}} \mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)\\ &= \sum_{\texttt{Vrot}}\left(\mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\left(\sum_{\texttt{Vtrans}} \mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)\right)\right).\end{aligned} \tag{6.10}$$

La marginalisation sur `Vtrans` fait disparaître le terme `Vtrans` du dénominateur :

$$\begin{aligned}&\sum_{\substack{\texttt{Vtrans}\\ \texttt{Vrot}}} \mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)\\ &= \sum_{\texttt{Vrot}}\Big(\mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)\Big).\end{aligned} \tag{6.11}$$

Enfin, la marginalisation sur `Vrot` fait disparaître le terme `Vrot` ; le dénominateur s'exprime finalement de la manière suivante :

$$\begin{aligned}&\sum_{\substack{\texttt{Vtrans}\\ \texttt{Vrot}}} \mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)\\ &= \mathrm{P}\big(\texttt{dir prox} \mid \mathit{Cp_Tourner}\big)\mathrm{P}\big(\texttt{vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big).\end{aligned} \tag{6.12}$$

En remplaçant le dénominateur par sa simplification, l'expression de la question devient :

$$\begin{aligned}&\mathrm{P}\big(\texttt{Vtrans Vrot} \mid \texttt{dir prox vtrans_c } \mathit{Cp_Evit}\big)\\ &= \frac{\mathrm{P}\big(\texttt{Vrot dir prox} \mid \mathit{Cp_Tourner}\big)}{\mathrm{P}\big(\texttt{dir prox} \mid \mathit{Cp_Tourner}\big)} \times \frac{\mathrm{P}\big(\texttt{Vtrans vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)}{\mathrm{P}\big(\texttt{vtrans_c} \mid \texttt{prox } \mathit{Cp_Ralentir}\big)}.\end{aligned} \tag{6.13}$$

Enfin, en appliquant la règle du produit, la question posée est résolue par l'expression suivante :

$$\begin{aligned}&\mathrm{P}(\texttt{Vtrans Vrot} \mid \texttt{dir prox vtrans_c } \mathit{Cp_Evit})\\&= \mathrm{P}(\texttt{Vrot} \mid \texttt{dir prox } \mathit{Cp_Tourner})\,\mathrm{P}(\texttt{Vtrans} \mid \texttt{prox vtrans_c } \mathit{Cp_Ralentir}).\end{aligned} \tag{6.14}$$

6.1.4.2 Résultat

Le résultat obtenu est très satisfaisant. Le robot ralentit lorsqu'il s'approche des obstacles. Cette vitesse plus faible améliore considérablement les performances obtenues avec le comportement d'évitement d'obstacle à vitesse constante, en particulier dans les cas difficiles où les obstacles réfléchissent peu l'infrarouge. De plus, ce comportement où les vitesses varient apparaît beaucoup plus « élégant » pour l'observateur humain.

Dans cet exemple, et pour la question que nous avons posée, cette forme de combinaison de descriptions peut apparaître superflue (voir relation (6.14)). En effet, nous avions spécifié `Vrot` et `Vtrans` de manière indépendante sous la forme de deux descriptions. La description décrivant la combinaison de ces deux comportements respecte le choix de commander indépendamment `Vrot` et `Vtrans`. Ainsi, connaissant les valeurs de `Dir`, `Prox` et `Vtrans_c`, un « résultat identique » serait obtenu en utilisant directement et indépendamment les deux descriptions, *Cp_Tourner* pour déterminer `Vrot` et *Cp_Ralentir* pour déterminer `Vtrans`.

En fait, l'indépendance entre les deux modalités est conditionnelle à la connaissance de la valeur de `Prox` (voir l'expression des connaissances préalables section 6.1.2.2). Dans le cas où nous ignorons la valeur de `Prox`, cette indépendance n'a plus lieu.

Par exemple, si nous cherchons à déterminer `Vrot` et `Vtrans` ne connaissant que les valeurs de `Dir` et `Vtrans_c`, la question à résoudre est la suivante :

$$\begin{aligned}&\mathrm{P}(\texttt{Vtrans Vrot} \mid \texttt{dir vtrans_c } \mathit{Cp_Evit})\\&= \frac{\mathrm{P}(\texttt{Vtrans Vrot dir vtrans_c} \mid \mathit{Cp_Evit})}{\mathrm{P}(\texttt{dir vtrans_c} \mid \mathit{Cp_Evit})}.\end{aligned} \tag{6.15}$$

La réponse à cette question sera de la forme :

$$\begin{aligned}&\mathrm{P}(\texttt{Vtrans Vrot} \mid \texttt{dir vtrans_c } \mathit{Cp_Evit})\\&= \frac{1}{Z}\sum_{\texttt{Prox}} \mathrm{P}(\texttt{Vrot dir Prox} \mid \mathit{Cp_Tourner})\,\mathrm{P}(\texttt{Vtrans vtrans_c} \mid \texttt{ Prox } \mathit{Cp_Ralentir}).\end{aligned} \tag{6.16}$$

Où Z indique la constante de normalisation suivante (obtenue par la marginalisation du numérateur sur les variables recherchées) :

$$Z = \sum_{\texttt{Vtrans Vrot}} \left(\sum_{\texttt{Prox}} \mathrm{P}(\texttt{Vrot dir Prox} \mid \mathit{Cp_Tourner})\,\mathrm{P}(\texttt{Vtrans vtrans_c} \mid \texttt{ Prox } \mathit{Cp_Ralentir}) \right). \tag{6.17}$$

Selon les connaissances de la description *Cp_Ralentir* (voir Chapitre 4 section 4.3.2), l'expression P(`Vtrans vtrans_c` | `Prox` *Cp_Ralentir*) se simplifie en :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Vtrans vtrans_c} \mid \texttt{Prox } \mathit{Cp_Ralentir}) \\
&= \frac{\mathrm{P}(\texttt{Vtrans vtrans_c Prox} \mid \mathit{Cp_Ralentir})}{\mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Ralentir})} \\
&= \frac{\mathrm{P}(\texttt{Vtrans} \mid \texttt{vtrans_c Prox } \mathit{Cp_Ralentir})\mathrm{P}(\texttt{vtrans_c} \mid \mathit{Cp_Ralentir})\mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Ralentir})}{\mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Ralentir})} \qquad (6.18) \\
&= \mathrm{P}(\texttt{Vtrans} \mid \texttt{vtrans_c Prox } \mathit{Cp_Ralentir})\mathrm{P}(\texttt{vtrans_c} \mid \mathit{Cp_Ralentir}) \\
&= \frac{1}{11}\mathrm{P}(\texttt{Vtrans} \mid \texttt{vtrans_c Prox } \mathit{Cp_Ralentir}).
\end{aligned}
$$

De la même manière, en appliquant les connaissances de la description *Cp_Tourner* (voir Chapitre 4 section 4.3.1), nous pouvons simplifier l'expression P(`Vrot dir Prox` | *Cp_Tourner*) :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Vrot dir Prox} \mid \mathit{Cp_Tourner}) \\
&= \mathrm{P}(\texttt{Vrot} \mid \texttt{dir Prox } \mathit{Cp_Tourner})\mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Tourner})\mathrm{P}(\texttt{dir} \mid \mathit{Cp_Tourner}) \qquad (6.19) \\
&= \frac{1}{16}\frac{1}{21}\mathrm{P}(\texttt{Vrot} \mid \texttt{dir Prox } \mathit{Cp_Tourner}).
\end{aligned}
$$

Finalement la connaissance sur `Vrot` et `Vtrans` ne connaissant que les valeurs de `Dir` et `Vtrans_c`, s'exprime de la manière suivante :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Vtrans Vrot} \mid \texttt{dir vtrans_c } \mathit{Cp_Evit}) \\
&= \frac{1}{Z'}\sum_{\texttt{Prox}} \mathrm{P}(\texttt{Vrot} \mid \texttt{dir Prox } \mathit{Cp_Tourner})\mathrm{P}(\texttt{Vtrans} \mid \texttt{vtrans_c Prox } \mathit{Cp_Ralentir}) \qquad (6.20)
\end{aligned}
$$

avec $Z' = Z \times 11 \times 16 \times 21$.

L'expression finale de cette question nous montre que lorsque l'on ignore la valeur de `Prox`, les connaissances sur `Vrot` et `Vtrans` ne peuvent plus être obtenue indépendamment.

Nous voyons ainsi l'intérêt qu'offre cette combinaison. En liant les deux descriptions, nous obtenons la possibilité de propager des connaissances d'une description à l'autre, pour les questions où les variables communes aux deux descriptions (ici `Prox`) ne sont pas connues avec certitude.

Dans la description que nous venons de spécifier, nous remarquons donc que la variable commune aux deux descriptions joue un rôle fondamental. C'est par son intermédiaire que les descriptions sont mises en relation. En effet, la connaissance sur cette variable va être influencée par les contraintes des dépendances où elle intervient dans chacune des descriptions.

Si nous cherchons à déterminer `Prox` selon la nouvelle description connaissant la valeur de toutes les autres variables, en suivant un raisonnement similaire, nous obtenons :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Prox} \mid \texttt{vrot dir vtrans vtrans_c}\ \mathit{Cp_Evit}) \\
&= \frac{\mathrm{P}(\texttt{vrot dir Prox vtrans vtrans_c} \mid \mathit{Cp_Evit})}{\mathrm{P}(\texttt{vrot dir vtrans vtrans_c} \mid \mathit{Cp_Evit})} \\
&= \frac{\mathrm{P}(\texttt{vrot dir Prox} \mid \mathit{Cp_Tourner})\mathrm{P}(\texttt{vtrans vtrans_c} \mid \texttt{Prox}\ \mathit{Cp_Ralentir})}{\sum_{\texttt{Prox}} \mathrm{P}(\texttt{vrot dir Prox} \mid \mathit{Cp_Tourner})\mathrm{P}(\texttt{vtrans vtrans_c} \mid \texttt{Prox}\ \mathit{Cp_Ralentir})} \\
&= \frac{1}{Z}\begin{vmatrix} \mathrm{P}(\texttt{vrot} \mid \texttt{dir Prox}\ \mathit{Cp_Tourner}) \\ \mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Tourner}) \\ \mathrm{P}(\texttt{vtrans} \mid \texttt{vtrans_c Prox}\ \mathit{Cp_Ralentir}). \end{vmatrix}
\end{aligned}
\tag{6.21}
$$

Où Z indique une constante de normalisation.

Nous remarquons que la connaissance sur la variable `Prox` sachant les valeurs de toutes les autres variables fait intervenir les deux dépendances élémentaires, P(`Vrot` | `Dir Prox` *Cp_Tourner*) et P(`Vtrans` | `Vtrans_c Prox` *Cp_Ralentir*), et l'a priori sur `Prox` exprimé dans la description *Cp_Tourner*.

Si, maintenant, nous cherchons à calculer l'a priori sur `Prox` selon la description *Cp_Evit*, P(`Prox` | *Cp_Evit*), nous obtenons :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Evit}) \\
&= \sum_{\substack{\texttt{Vrot} \\ \texttt{Dir} \\ \texttt{Vtrans} \\ \texttt{Vtrans_c}}} \mathrm{P}(\texttt{Vrot Dir Prox Vtrans Vtrans_c} \mid \mathit{Cp_Evit}) \\
&= \sum_{\substack{\texttt{Vrot} \\ \texttt{Dir} \\ \texttt{Vtrans} \\ \texttt{Vtrans_c}}} \begin{vmatrix} \mathrm{P}(\texttt{Vrot Dir Prox} \mid \mathit{Cp_Tourner}) \\ \mathrm{P}(\texttt{Vtrans Vtrans_c} \mid \texttt{Prox}\ \mathit{Cp_Ralentir}) \end{vmatrix} \\
&= \mathrm{P}(\texttt{Prox} \mid \mathit{Cp_Tourner}).
\end{aligned}
\tag{6.22}
$$

Nous remarquons alors que le résultat ne prend en compte que l'a priori sur `Prox` exprimé dans la description *Cp_Ralentir*. Ceci est le résultat du choix de la décomposition retenue. D'autres décompositions sont envisageables, chacune d'elles exprime un a priori différent sur la variable `Prox`.

Nous notons *Cp_Evit'* les connaissances associées à la décomposition suivante :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid \mathit{Cp_Evit'}) \\
&= \begin{vmatrix} \mathrm{P}(\texttt{Vtrans Vtrans_c Prox} \mid \mathit{Cp_Ralentir}) \\ \mathrm{P}(\texttt{Vrot Dir} \mid \texttt{Prox}\ \mathit{Cp_Tourner}). \end{vmatrix}
\end{aligned}
\tag{6.23}
$$

Cette nouvelle description fournit une réponse identique pour le calcul des ordres moteurs `Vtrans` et `Vrot` connaissant les valeurs des autres variables (`Dir`, `Prox`, `Vtrans_c`). Mais cette fois ci, l'a priori sur la variable `Prox`, P(`Prox` | *Cp_Evit'*), est différent et correspond à celui exprimé dans la description *Cp_Ralentir* :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Prox} \mid Cp_Evit') \\
&= \sum_{\substack{\texttt{Vrot}\\ \texttt{Dir}\\ \texttt{Vtrans}\\ \texttt{Vtrans_c}}} \mathrm{P}(\texttt{Vrot Dir Prox Vtrans Vtrans_c} \mid Cp_Evit') \\
&= \sum_{\substack{\texttt{Vrot}\\ \texttt{Dir}\\ \texttt{Vtrans}\\ \texttt{Vtrans_c}}} \left| \begin{array}{l} \mathrm{P}(\texttt{Vtrans Vtrans_c Prox} \mid Cp_Ralentir) \\ \mathrm{P}(\texttt{Vrot Dir} \mid \texttt{Prox}\ Cp_Tourner) \end{array} \right. \\
&= \mathrm{P}(\texttt{Prox} \mid Cp_Ralentir).
\end{aligned}
\tag{6.24}
$$

Une autre possibilité serait d'exprimer un nouvel a priori sur `Prox` dans la description de la combinaison. Nous notons *Cp_Evit''* les connaissances préalables associées à cette nouvelle description :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid Cp_Evit'') \\
&= \left| \begin{array}{l} \mathrm{P}(\texttt{Prox} \mid Cp_Evit'') \\ \mathrm{P}(\texttt{Vrot Dir} \mid \texttt{Prox}\ \ Cp_Tourner) \\ \mathrm{P}(\texttt{Vtrans Vtrans_c} \mid \texttt{Prox}\ Cp_Ralentir). \end{array} \right.
\end{aligned}
\tag{6.25}
$$

La combinaison de ces deux comportements se résume au produit de trois distributions élémentaires : les deux distributions associées chacune à un comportement et une distribution portant sur la variable `Prox`. Ici, c'est au programmeur de spécifier le nouvel a priori sur Prox, P(`Prox` | *Cp_Evit''*).

Enfin, nous pouvons choisir une dernière possibilité pour exprimer cette combinaison, de manière à obtenir un a priori sur `Prox` qui soit le reflet des connaissances exprimées dans les deux descriptions *Cp_Tourner* et *Cp_Ralentir*. Nous notons *Cp_Evit'''* les connaissances préalables associées à la décomposition suivante :

$$
\begin{aligned}
&\mathrm{P}(\texttt{Dir Prox Vtrans_c Vtrans Vrot} \mid Cp_Evit''') \\
&= \frac{\mathrm{P}(\texttt{Vrot Dir Prox} \mid Cp_Tourner)\,\mathrm{P}(\texttt{Vtrans Vtrans_c Prox} \mid Cp_Ralentir)}{\mathrm{P}(\texttt{Prox} \mid Cp_Evit''')}.
\end{aligned}
\tag{6.26}
$$

La connaissance a priori sur `Prox` dans cette nouvelle description s'obtient par marginalisation de la distribution conjointe et s'exprime comme une moyenne géométrique des a priori sur `Prox` exprimées dans les deux descriptions *Cp_Tourner* et *Cp_Ralentir* :

$$
\mathrm{P}(\texttt{Prox} \mid Cp_Evit''') = \sqrt{\mathrm{P}(\texttt{Prox} \mid Cp_Tourner)\,\mathrm{P}(\texttt{Prox} \mid Cp_Ralentir)}. \tag{6.27}
$$

Nous détaillons plus bas le raisonnement suivi pour obtenir cette décomposition et ce dernier résultat (voir (6.32) et suivant).

En résumé, la description que nous venons de spécifier est une forme extrêmement simple de combinaison de deux comportements associés à des modalités motrices indépendantes (par exemple M1 et M2) ayant une modalité sensorielle en commun (par exemple S). Soit P(M1 S | *C1*) et P(M2 S | *C2*) les deux descriptions à combiner.

Nous notons P(M1 M2 S | *C*) la description décrivant la combinaison. Les connaissances qui doivent être fournies pour décrire ce type de combinaison se résument à :

- D'une part l'expression d'une hypothèse d'indépendance conditionnelle :

P(M1 | ~~M2~~ S *C*) = P(M1 | S *C*) qui est équivalent à l'hypothèse symétrique P(M2 | ~~M1~~ S *C*) = P(M2 | S *C*)

- D'autre part, le choix d'un a priori pour la variable commune S, parmis les quatre alternatives suivantes :

1) Soit P(S | *C*) est spécifié complètement au sein de la nouvelle description

$$\mathrm{P}(\mathtt{M1\ M2\ S} \mid C) = \mathrm{P}(\mathtt{M1} \mid \mathtt{S}\ C1)\,\mathrm{P}(\mathtt{M2} \mid \mathtt{S}\ C2)\,\mathrm{P}(\mathtt{S} \mid C). \tag{6.28}$$

2) Soit il est obtenu à partir de la description *C1* : P(S | *C*) = P(S | *C1*)

$$\mathrm{P}(\mathtt{M1\ M2\ S} \mid C) = \mathrm{P}(\mathtt{M1\ S} \mid C1)\,\mathrm{P}(\mathtt{M2} \mid \mathtt{S}\ C2). \tag{6.29}$$

3) Soit il est obtenu à partir de la description *C2* : P(S | *C*) = P(S | *C2*)

$$\mathrm{P}(\mathtt{M1\ M2\ S} \mid C) = \mathrm{P}(\mathtt{M2\ S} \mid C2)\,\mathrm{P}(\mathtt{M1} \mid \mathtt{S}\ C1). \tag{6.30}$$

4) Soit il est une moyenne géométrique des deux :

$$\mathrm{P}(\mathtt{M1\ M2\ S} \mid C) = \frac{\mathrm{P}(\mathtt{M2\ S} \mid C2)\,\mathrm{P}(\mathtt{M1\ S} \mid C1)}{\sqrt{\mathrm{P}(\mathtt{S} \mid C2)\,\mathrm{P}(\mathtt{S} \mid C1)}}. \tag{6.31}$$

L'expression de la relation (6.31) s'obtient en suivant le raisonnement suivant :

$$\begin{aligned}
\mathrm{P}(\mathtt{M1\ M2\ S} \mid C) &= \mathrm{P}(\mathtt{M1\ S} \mid C)\,\mathrm{P}(\mathtt{M2} \mid \mathtt{M1\ S}\ C) && \text{par la règle du produit} \\
&= \mathrm{P}(\mathtt{M1\ S} \mid C)\,\mathrm{P}(\mathtt{M2} \mid \mathtt{S}\ C) && \text{par hypothèse d' indépendance conditionnelle} \\
&= \mathrm{P}(\mathtt{M1\ S} \mid C)\,\frac{\mathrm{P}(\mathtt{M2\ S} \mid C)}{\mathrm{P}(\mathtt{S} \mid C)} && \text{par la règle du produit} \\
&= \frac{\mathrm{P}(\mathtt{M1\ S} \mid C)\,\mathrm{P}(\mathtt{M2\ S} \mid C)}{\mathrm{P}(\mathtt{S} \mid C)}.
\end{aligned} \tag{6.32}$$

En décidant d'associer les distributions conjointes des descriptions C1 et C2 à leur forme respective dans cette décomposition nous obtenons :

$$\mathrm{P}(\mathtt{M1\ M2\ S} \mid C) = \frac{\mathrm{P}(\mathtt{M1\ S} \mid C1)\,\mathrm{P}(\mathtt{M2\ S} \mid C2)}{\mathrm{P}(\mathtt{S} \mid C)}. \tag{6.33}$$

Il reste à déterminer P(S | C), qui s'obtient par marginalisation :

$$\begin{aligned} \mathrm{P}(\mathtt{S} \mid C) &= \sum_{\substack{\mathtt{M1} \\ \mathtt{M2}}} \mathrm{P}(\mathtt{M1}\ \mathtt{M2}\ \mathtt{S} \mid C) \\ &= \sum_{\substack{\mathtt{M1} \\ \mathtt{M2}}} \left(\frac{\mathrm{P}(\mathtt{M1}\ \mathtt{S} \mid C1)\mathrm{P}(\mathtt{M2}\ \mathtt{S} \mid C2)}{\mathrm{P}(\mathtt{S} \mid C)} \right). \end{aligned} \tag{6.34}$$

Le terme P(S | *C*) est indépendant de M1 et de M2, il peut être sorti de la somme :

$$\begin{aligned} \mathrm{P}(\mathtt{S} \mid C) &= \frac{1}{\mathrm{P}(\mathtt{S} \mid C)} \sum_{\substack{\mathtt{M1} \\ \mathtt{M2}}} \mathrm{P}(\mathtt{M1}\ \mathtt{S} \mid C1)\mathrm{P}(\mathtt{M2}\ \mathtt{S} \mid C2) \\ &= \frac{1}{\mathrm{P}(\mathtt{S} \mid C)} \sum_{\mathtt{M1}} \left(\mathrm{P}(\mathtt{M1}\ \mathtt{S} \mid C1) \left(\sum_{\mathtt{M2}} \mathrm{P}(\mathtt{M2}\ \mathtt{S} \mid C2) \right) \right) \\ &= \frac{1}{\mathrm{P}(\mathtt{S} \mid C)} \sum_{\mathtt{M1}} \left(\mathrm{P}(\mathtt{M1}\ \mathtt{S} \mid C1)\mathrm{P}(\mathtt{S} \mid C2) \right) \\ &= \frac{1}{\mathrm{P}(\mathtt{S} \mid C)} \mathrm{P}(\mathtt{S} \mid C2) \sum_{\mathtt{M1}} \mathrm{P}(\mathtt{M1}\ \mathtt{S} \mid C1) \\ &= \frac{1}{\mathrm{P}(\mathtt{S} \mid C)} \mathrm{P}(\mathtt{S} \mid C2)\mathrm{P}(\mathtt{S} \mid C1). \end{aligned} \tag{6.35}$$

Et finalement, l'a priori sur la variable S est :

$$\mathrm{P}(\mathtt{S} \mid C) = \sqrt{\mathrm{P}(\mathtt{S} \mid C2)\mathrm{P}(\mathtt{S} \mid C1)}. \tag{6.36}$$

6.2 Se diriger vers une source lumineuse en évitant les obstacles

6.2.1 Objectifs et protocole expérimental

Nous disposons de deux comportements réactifs, évitement d'obstacle et phototaxie. L'objectif de cette expérience est d'illustrer une solution pour combiner ces deux comportements afin d'obtenir un nouveau comportement dans lequel le Khepera va vers la lumière tout en évitant les obstacles. Il s'agit bien d'une combinaison où les deux comportements sont « mixés » l'un avec l'autre, à la différence d'une stratégie consistant à « basculer » sans nuance d'un comportement à l'autre.

Par la suite, ce nouveau comportement sera utilisé par le Khepera pour rentrer à sa base, matérialisée par une source lumineuse. Le comportement final recherché doit donc mener le Khepera à se diriger vers la source lumineuse lorsqu'il est en espace libre, et à éviter les obstacles lorsqu'il s'en rapproche.

Dans cette expérience, il n'y a pas à proprement parler d'identification expérimentale. Nous réutilisons les deux descriptions définies préalablement ; leur combinaison sera spécifiée a priori.

6.2.2 Spécification

Nous disposons de deux comportements de base « évitement d'obstacle » représenté par P(Vrot Dir Prox | *Cp_Evit*) et « phototaxie » représenté par P(Vrot Lum | *Cp_Photo*) :

Comportement phototaxie :

$$\begin{aligned} &\mathrm{P}(\texttt{Vrot Lum} \mid Cp_Photo) \\ &= \mathrm{P}(\texttt{Vrot} \mid \texttt{Lum}\ Cp_Photo)\,\mathrm{P}(\texttt{Lum} \mid Cp_Photo)\ ; \end{aligned} \tag{6.37}$$

Comportement d' évitement d' obstacle :

$$\begin{aligned} &\mathrm{P}(\texttt{Vrot Dir Prox} \mid Cp_Evit) \\ &= \mathrm{P}(\texttt{Vrot} \mid \texttt{Dir Prox}\ Cp_Evit)\,\mathrm{P}(\texttt{Dir} \mid Cp_Evit)\,\mathrm{P}(\texttt{Prox} \mid Cp_Evit). \end{aligned}$$

6.2.2.1 Choix des variables

Les variables qui entrent en jeu pour la description sont naturellement les 4 variables associées aux deux comportements initiaux, Vrot, Dir, Prox, Lum. Pour mixer les deux comportements, nous créons une variable supplémentaire H.

H est une variable pouvant prendre deux valeurs et qui représente les alternatives : [H = e] pour « évitement d'obstacle » et [H = p] pour « phototaxie ». Ainsi, lorsque la variable H a pour valeur e, nous souhaitons que le Khepera ait un comportement d'évitement d'obstacle spécifié par les connaissances *Cp_Evit* et réciproquement, le comportement spécifié par *Cp_Photo* pour [H = p].

Variables pertinentes

- **Dir** : domaine {-10, -9,…,.+10}, $k_{\texttt{Dir}} = 21$;
- **Prox** : domaine {0, 1,…, 15}, $k_{\texttt{Prox}} = 16$;
- **Lum** : domaine {-170, -90, -45, -10, +10, +45, +90, +170}, $k_{\texttt{Lum}} = 8$;
- **Vrot** : domaine {-10, -9,…, +10}, $k_{\texttt{Vrot}} = 21$;
- **H** : domaine {p, e}, $k_{\texttt{H}} = 2$.

6.2.2.2 Connaissances préalables de dépendance

La description du modèle porte sur les variables Vrot, Dir, Prox, Lum et H, et correspond à la distribution conjointe :

$$\mathrm{P}(\texttt{Lum Dir Prox H Vrot} \mid D\ C). \tag{6.38}$$

Dans un premier temps, nous décomposons la distribution conjointe par l'application successive de la règle du produit **[R1]** et obtenons ainsi cinq distributions élémentaires :

$$\begin{aligned}&\mathrm{P}(\texttt{Lum Dir Prox H Vrot} \mid D\ C)\\ &= \begin{vmatrix}\mathrm{P}(\texttt{Lum} \mid D\ C)\\ \mathrm{P}(\texttt{Dir} \mid \texttt{Lum}\ D\ C)\\ \mathrm{P}(\texttt{Prox} \mid \texttt{Dir Lum}\ D\ C)\\ \mathrm{P}(\texttt{H} \mid \texttt{Prox Dir Lum}\ D\ C)\\ \mathrm{P}(\texttt{Vrot} \mid \texttt{H Prox Dir Lum}\ D\ C).\end{vmatrix}\end{aligned} \tag{6.39}$$

Puis nous exprimons des hypothèses d'indépendance conditionnelle.

Nous choisissons de considérer que la direction des obstacles est indépendante de la direction de la source lumineuse :

$$\mathrm{P}(\texttt{Dir} \mid \texttt{Lum}\ D\ C) = \mathrm{P}(\texttt{Dir} \mid D\ C). \tag{6.40}$$

De même, nous considérons la proximité des obstacles indépendante de leur direction et de la direction de la source lumineuse :

$$\mathrm{P}(\texttt{Prox} \mid \texttt{Dir Lum}\ D\ C) = \mathrm{P}(\texttt{Prox} \mid D\ C). \tag{6.41}$$

Enfin, nous spécifions que le choix entre évitement d'obstacle ([H = e]) et phototaxie ([H = p]) ne dépend que de la proximité des obstacles, Prox :

$$\mathrm{P}(\texttt{H} \mid \texttt{Prox Dir Lum}\ D\ C) = \mathrm{P}(\texttt{H} \mid \texttt{Prox}\ D\ C). \tag{6.42}$$

Finalement, la décomposition complète obtenue est :

$$\begin{aligned}&\mathrm{P}(\texttt{Lum Dir Prox H Vrot} \mid D\ C)\\ &= \begin{vmatrix}\mathrm{P}(\texttt{Lum} \mid D\ C)\mathrm{P}(\texttt{Dir} \mid D\ C)\mathrm{P}(\texttt{Prox} \mid D\ C)\\ \mathrm{P}(\texttt{H} \mid \texttt{Prox}\ D\ C)\\ \mathrm{P}(\texttt{Vrot} \mid \texttt{H Prox Dir Lum}\ D\ C).\end{vmatrix}\end{aligned} \tag{6.43}$$

6.2.2.3 Connaissances préalables d'observation

Nous ne privilégions pas certaines situations sensorielles vis-à-vis d'autres (la source lumineuse peut être dans n'importe quelle direction par rapport au robot, et de même pour les obstacles) et nous ne souhaitons pas observer leur répartition expérimentale :

$$\begin{aligned}&\mathrm{P}([\texttt{Lum} = \texttt{lum}_i] \mid D\ C) = \mathrm{P}([\texttt{Lum} = \texttt{lum}_i] \mid C) = \mathbf{U} = \frac{1}{8},\\ &\mathrm{P}([\texttt{Prox} = \texttt{prox}_j] \mid D\ C) = \mathrm{P}([\texttt{Prox} = \texttt{prox}_j] \mid C) = \mathbf{U} = \frac{1}{16},\\ &\mathrm{P}([\texttt{Dir} = \texttt{dir}_k] \mid D\ C) = \mathrm{P}([\texttt{Dir} = \texttt{dir}_k] \mid C) = \mathbf{U} = \frac{1}{21}.\end{aligned} \tag{6.44}$$

Nous allons spécifier la relation entre la variable H et Prox fonctionnellement, en indiquant également que cette distribution est fixée a priori et donc n'évoluera pas avec l'expérience :

$$\begin{aligned} \mathrm{P}\big([\mathtt{H}=\mathtt{e}] \mid \mathtt{Prox}\ D\ C\big) &= \mathrm{P}\big([\mathtt{H}=\mathtt{e}] \mid \mathtt{Prox}\ C\big) = \mathbf{f}(\mathtt{Prox}) \\ \mathrm{P}\big([\mathtt{H}=\mathtt{p}] \mid \mathtt{Prox}\ D\ C\big) &= \mathrm{P}\big([\mathtt{H}=\mathtt{p}] \mid \mathtt{Prox}\ C\big) = 1-\mathbf{f}(\mathtt{Prox}). \end{aligned} \tag{6.45}$$

Il reste à spécifier la distribution fixant la probabilité de `Vrot`. Pour cela, nous allons tout simplement indiquer, selon la valeur de la variable `H`, la description correspondante : *Cp_Photo* pour [`H` = `p`] et *Cp_Evit* pour [`H` = `e`] :

$$\begin{aligned} \mathrm{P}\big(\mathtt{Vrot} \mid [\mathtt{H}=\mathtt{e}]\ \mathtt{Prox}\ \mathtt{Dir}\ \mathtt{Lum}\ D\ C\big) &= \mathrm{P}\big(\mathtt{Vrot} \mid \mathtt{Prox}\ \mathtt{Dir}\ Cp_Evit\big) \\ \mathrm{P}\big(\mathtt{Vrot} \mid [\mathtt{H}=\mathtt{p}]\ \mathtt{Prox}\ \mathtt{Dir}\ \mathtt{Lum}\ D\ C\big) &= \mathrm{P}\big(\mathtt{Vrot} \mid \mathtt{Lum}\ Cp_Photo\big).. \end{aligned} \tag{6.46}$$

6.2.3 Identification

Il reste enfin à spécifier la forme fonctionnelle associée à $\mathrm{P}(\mathtt{H} \mid \mathtt{Prox}\ \ C)$.

Pour P([`H`=`e`] | `Prox` *C*) nous utilisons une fonction sigmoïde en `Prox` avec $\alpha = 9$ et $\beta = 0.25$ (Figure 6-1) :

$$\begin{gathered} \mathrm{P}\big([\mathtt{H}=\mathtt{e}] \mid \mathtt{Prox}\ C\big) = \mathrm{sigmoid}_{\alpha,\beta}(\mathtt{Prox}) \\ \text{avec } \alpha = 9 \text{ et } \beta = 0{,}25. \end{gathered} \tag{6.47}$$

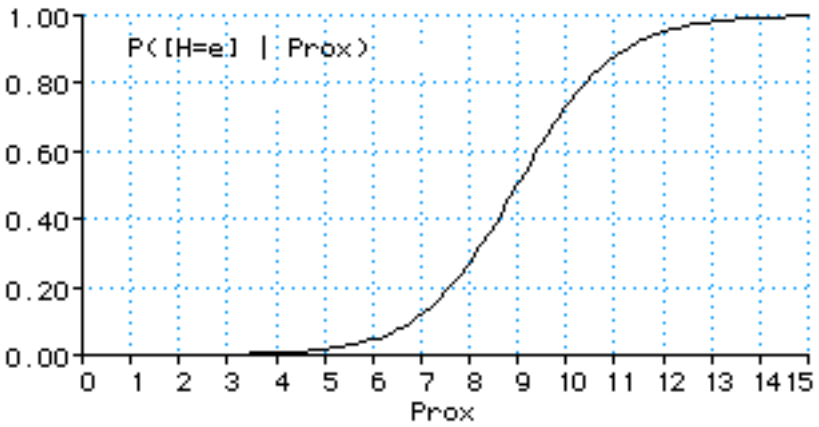


Figure 6-1 — La dépendance P([`H`=`e`] | `Prox` *C*) spécifiée comme fonction de `Prox` par une sigmoïde.

La valeur de P([`H`=`p`] | `Prox` *C*) est obtenue par normalisation :

$$\mathrm{P}\big([\mathtt{H}=\mathtt{p}] \mid \mathtt{Prox}\ C\big) = 1 - \mathrm{P}\big([\mathtt{H}=\mathtt{e}] \mid \mathtt{Prox}\ C\big). \tag{6.48}$$

Par la suite, nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_E⊕P* :

$$\begin{aligned}&\mathrm{P}(\texttt{Lum Dir Prox H Vrot} \mid Cp_E \oplus P)\\ &= \begin{vmatrix}\mathrm{P}(\texttt{Lum} \mid Cp_E \oplus P)\mathrm{P}(\texttt{Dir} \mid Cp_E \oplus P)\mathrm{P}(\texttt{Prox} \mid Cp_E \oplus P)\\ \mathrm{P}(\texttt{H} \mid \texttt{Prox}\ Cp_E \oplus P)\\ \mathrm{P}(\texttt{Vrot} \mid \texttt{H Prox Dir Lum}\ Cp_E \oplus P).\end{vmatrix}\end{aligned} \tag{6.49}$$

6.2.4 Utilisation

6.2.4.1 Programme et question probabiliste

Nous avons utilisé cette description pour piloter le Khepera de manière à ce qu'il se dirige ver une source lumineuse tout en évitant les obstacles. Ce comportement est obtenu par la requête suivante :

1/ Descriptions utilisées :

Cp_E⊕P, Cp_Photo, Cp_Evit

2/ Requête :

E_q : {Vrot}

E_c : {Dir, Prox, Lum}

E_i : {H}

Question : P(Vrot | Dir Prox Lum *Cp_E⊕P*)

3/ Décision :

Tirage aléatoire

Nous allons détailler la résolution de la question. À nouveau, nous notons les variables dont les valeurs sont connues (ici `Dir`, `Prox` et `Lum`) en minuscule (`dir`, `prox` et `lum`)

Ainsi, la question que l'on pose à la description consiste à calculer l'expression suivante :

$$\begin{aligned}&\mathrm{P}(\texttt{Vrot} \mid \texttt{lum dir prox}\ Cp_E \oplus P)\\ &= \frac{\mathrm{P}(\texttt{Vrot lum dir prox} \mid Cp_E \oplus P)}{\mathrm{P}(\texttt{lum dir prox} \mid Cp_E \oplus P)}.\end{aligned} \tag{6.50}$$

La résolution de cette question s'obtient par marginalisation du numérateur sur la variable `H`, et la marginalisation du dénominateur sur les variables `H` et `Vrot` :

$$\begin{aligned}&\mathrm{P}(\texttt{Vrot} \mid \texttt{lum dir prox}\ Cp_E \oplus P)\\ &= \frac{\sum_{\texttt{H}} \mathrm{P}(\texttt{Vrot H lum dir prox} \mid Cp_E \oplus P)}{\sum_{\substack{\texttt{Vrot}\\ \texttt{H}}} \mathrm{P}(\texttt{Vrot H lum dir prox} \mid Cp_E \oplus P)}.\end{aligned} \tag{6.51}$$

En remplaçant l'expression de la distribution conjointe par sa décomposition, le dénominateur se simplifie en :

$$\begin{aligned}&\sum_{\substack{\texttt{Vrot}\\ \texttt{H}}} \mathrm{P}(\texttt{Vrot H lum dir prox} \mid Cp_E \oplus P)\\ &= \sum_{\substack{\texttt{Vrot}\\ \texttt{H}}} \left| \begin{array}{l} \mathrm{P}(\texttt{lum} \mid Cp_E \oplus P)\mathrm{P}(\texttt{dir} \mid Cp_E \oplus P)\mathrm{P}(\texttt{prox} \mid Cp_E \oplus P)\\ \mathrm{P}(\texttt{H} \mid \texttt{prox}\ Cp_E \oplus P)\\ \mathrm{P}(\texttt{Vrot} \mid \texttt{H lum dir prox}\ Cp_E \oplus P). \end{array} \right.\end{aligned} \tag{6.52}$$

Les a priori sur les variables `dir`, `prox` et `lum` ne dépendent ni de `Vrot`, ni de `H` et, par conséquent, ils peuvent être mis en facteur de la somme :

$$\begin{aligned}&\sum_{\substack{\texttt{Vrot}\\ \texttt{H}}} \mathrm{P}(\texttt{Vrot H lum dir prox} \mid Cp_E \oplus P)\\ &= \left| \begin{array}{l} \mathrm{P}(\texttt{lum} \mid Cp_E \oplus P)\mathrm{P}(\texttt{dir} \mid Cp_E \oplus P)\mathrm{P}(\texttt{prox} \mid Cp_E \oplus P)\\ \sum_{\texttt{H}} \left(\mathrm{P}(\texttt{H} \mid \texttt{prox}\ Cp_E \oplus P) \sum_{\texttt{Vrot}} \mathrm{P}(\texttt{Vrot} \mid \texttt{H lum dir prox}\ Cp_E \oplus P) \right). \end{array} \right.\end{aligned} \tag{6.53}$$

Le dénominateur se résume au produit des trois a priori uniformes sur les variables `Lum`, `Dir` et `Prox`.

En appliquant un raisonnement similaire, le numérateur s'exprime ainsi :

$$\begin{aligned}&\sum_{\texttt{H}} \mathrm{P}(\texttt{Vrot H lum dir prox} \mid Cp_E \oplus P)\\ &= \left| \begin{array}{l} \mathrm{P}(\texttt{lum} \mid Cp_E \oplus P)\mathrm{P}(\texttt{dir} \mid Cp_E \oplus P)\mathrm{P}(\texttt{prox} \mid Cp_E \oplus P)\\ \sum_{\texttt{H}} \left(\mathrm{P}(\texttt{H} \mid \texttt{prox}\ Cp_E \oplus P)\mathrm{P}(\texttt{Vrot} \mid \texttt{H lum dir prox}\ Cp_E \oplus P) \right). \end{array} \right.\end{aligned} \tag{6.54}$$

En remplaçant dans (6.51) le dénominateur et le numérateur par (6.53) et (6.54), les a priori sur les valeurs des variables `lum`, `dir`, `prox` disparaissent et la question se résume à l'expression suivante :

$$\begin{aligned}&\mathrm{P}(\texttt{Vrot} \mid \texttt{lum dir prox}\ Cp_E \oplus P)\\ &= \sum_{\texttt{H}} \left(\mathrm{P}(\texttt{Vrot} \mid \texttt{H lum dir prox}\ Cp_E \oplus P)\mathrm{P}(\texttt{H} \mid \texttt{prox}\ Cp_E \oplus P) \right).\end{aligned} \tag{6.55}$$

Finalement, la distribution de probabilité sur `Vrot` connaissant les valeurs des variables `Dir`, `Prox` et `Lum` s'exprime sous la forme de la somme pondérée :

$$\mathrm{P}(\texttt{Vrot} \mid \texttt{lum dir prox } Cp_E \oplus P) = \begin{vmatrix} \mathrm{P}([\texttt{H} = \texttt{e}] \mid \texttt{prox } Cp_E \oplus P)\mathrm{P}(\texttt{Vrot} \mid \texttt{dir prox } Cp_Evit) \\ + \\ (1 - \mathrm{P}([\texttt{H} = \texttt{e}] \mid \texttt{prox } Cp_E \oplus P))\mathrm{P}(\texttt{Vrot} \mid \texttt{lum } Cp_Photo). \end{vmatrix} \quad (6.56)$$

6.2.4.2 Résultat

Dans cette expérience, nous avons obtenu un comportement global phototaxie avec évitement d'obstacle très acceptable. Loin des obstacles, le robot se dirige résolument vers la lumière. En présence d'obstacles très proches, le robot ne prête aucune attention à la direction de la source lumineuse et évite bien ces obstacles. En présence d'obstacles moyennement proche (juste détectables sans être dangereux), la lumière peut inciter le robot à choisir un sens d'évitement inhabituel qui temporairement le fait se tourner vers l'obstacle. La Figure 6-2 illustre une telle situation.

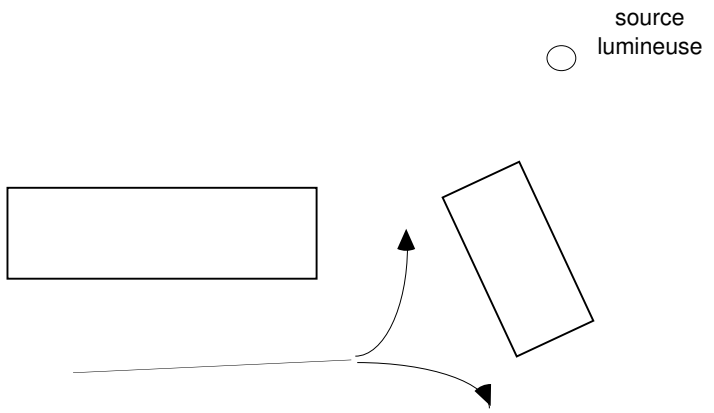


Figure 6-2 — Combinaison évitement d'obstacle et phototaxie.
Un cas (très schématisé) où l'évitement d'obstacle pur conduit à tourner à droite, mais où la composition avec l'attirance vers la lumière permet de trouver la solution de tourner à gauche.

Figure 6-3 —Combinaison des comportements « phototaxie » et « évitement d'obstacle ».
La source lumineuse se trouve en haut de l'image (indiquée par la flèche). Le Khepera a tout d'abord évité le premier obstacle en bois, puis s'est dirigé vers la source en longeant l'obstacle triangulaire.

Nous avons testé cette combinaison dans un environnement fortement encombré en présence d'une source lumineuse suffisamment forte (voir par exemple Figure 6-3). Quel que soit l'endroit d'où partait le robot, il est parvenu à remonter le gradient lumineux. Très rarement, il est resté « coincé » dans des minima locaux (obstacle en forme de U s'interposant entre le robot et la source lumineuse). Cette faculté de toujours retrouver le chemin menant à la source lumineuse nous a suggéré d'utiliser cela comme un comportement de « retour au gîte » pour l'application à la tâche de surveillance que nous décrivons au Chapitre 10.

Les figures suivantes présentent les distributions obtenues dans le cas d'une source lumineuse à gauche du robot (Lum = -90) pour différentes situations sensorielles pour `Dir` et `Prox`. Pour chaque figure, la distribution en haut à gauche correspond au comportement phototaxie, P(`Vrot` | `Lum` *Cp_Photo*) ; en haut à droite, le comportement d'évitement d'obstacle, P(`Vrot` | `Dir Prox` *Cp_Evit*) ; en bas à gauche la distribution sur la variable de combinaison H, P(`H` | `Prox` *Cp_E⊕P*) et en bas à droite le résultat de la combinaison P(`Vrot` | `Dir Prox Lum` *Cp_E⊕P*)

La Figure 6-4 illustre le résultat obtenu pour un obstacle proche à gauche du robot (Prox = 10 et Dir = -5). Les deux comportements aboutissent à des ordres moteurs antagonistes. La proximité de l'obstacle inhibe le comportement phototaxie. Cette inhibition décroît lorsque l'obstacle est plus éloigné du robot (`Prox` = 8) et autorise le comportement phototaxie (Figure 6-5). En présence d'un obstacle du côté opposé à la source lumineuse, le comportement d'évitement se trouve renforcé par le comportement phototaxie (Figure 6-6).

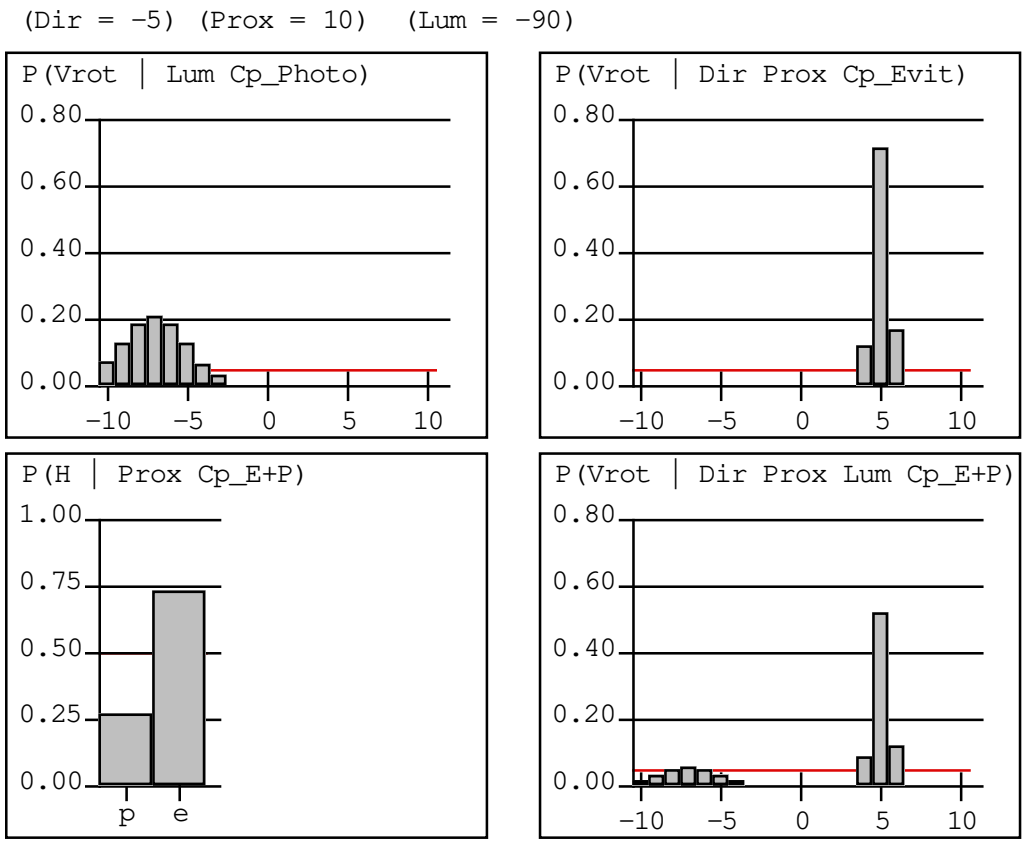


Figure 6-4 — Combinaison phototaxie et évitement d'obstacle : `Lum` = -90, `Dir` = -5, `Prox` = 10.

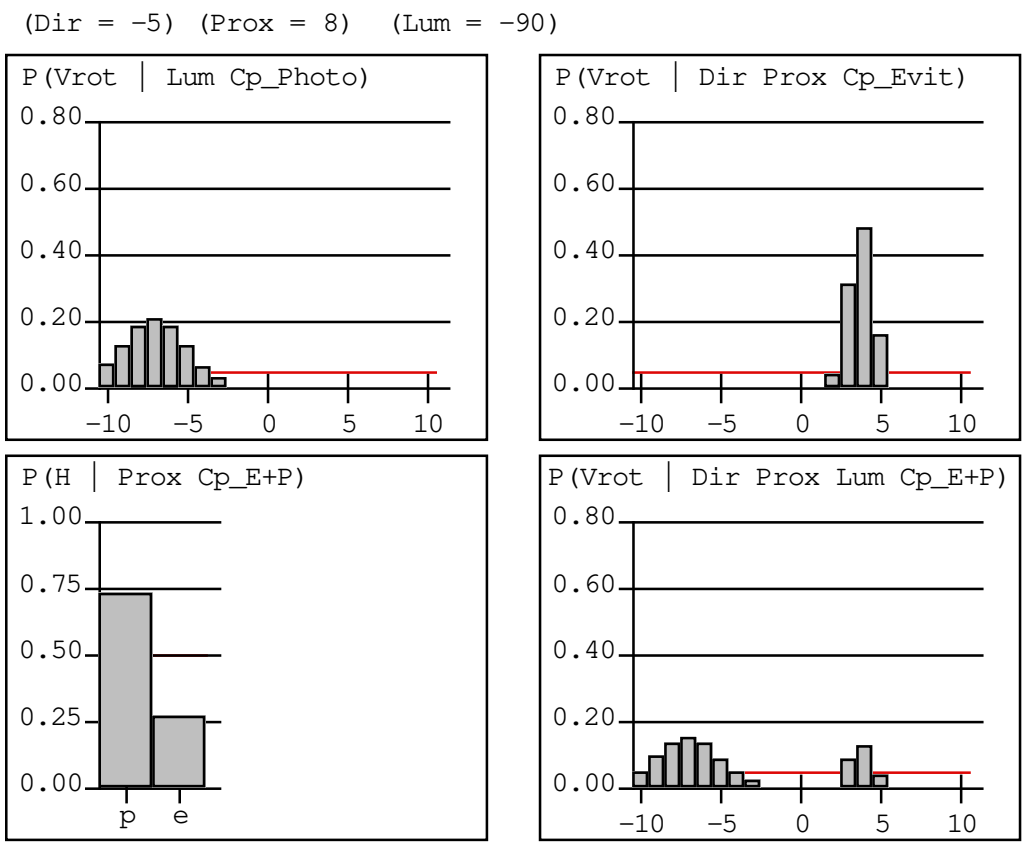


Figure 6-5 — Combinaison phototaxie et évitement d'obstacle : `Lum` = -90, `Dir` = -5, `Prox` = 10.

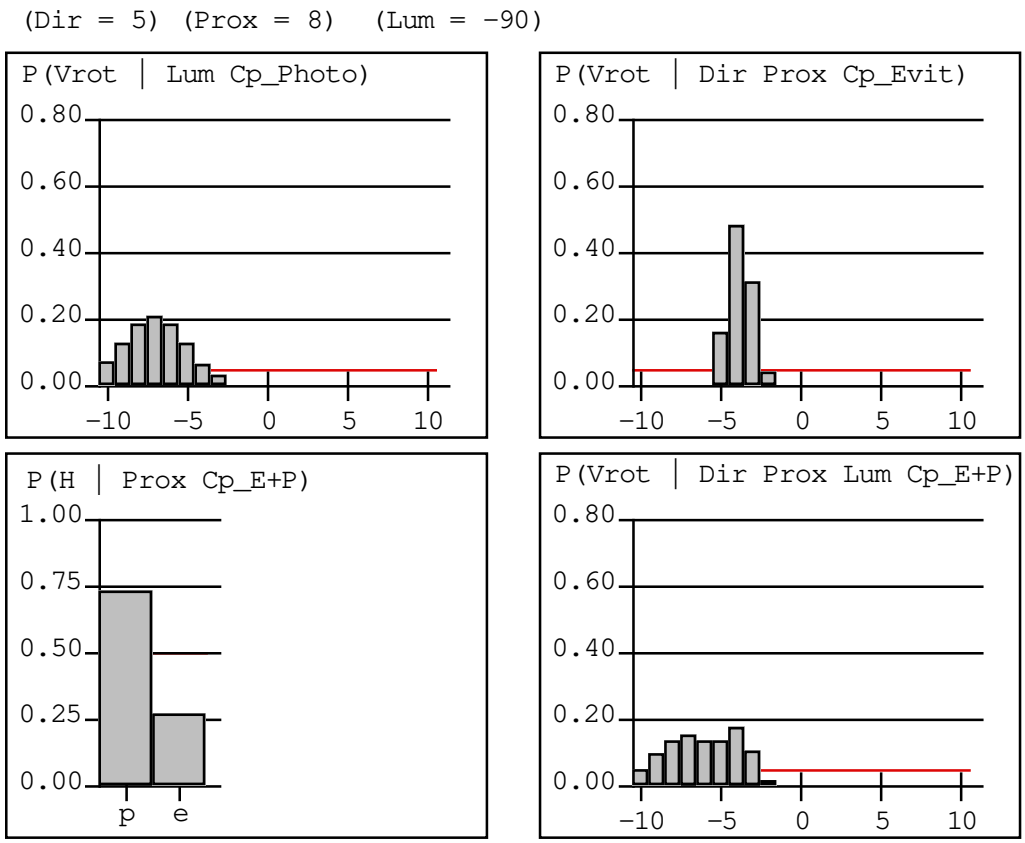


Figure 6-6 — Combinaison phototaxie et évitement d'obstacle : `Lum` = -90, `Dir` = 5, `Prox` = 8.

6.3 Enseignements et discussion

Le principal enseignement à retenir de ces deux expériences est que le formalisme probabiliste permet d'aisément et rigoureusement combiner entre eux des comportements acquis séparément. Ceci nous semble être un gros avantage de l'approche probabiliste par rapport à d'autres approches permettant l'acquisition de comportements réactifs (réseaux de neurones, subsumption architecture, etc.). Les probabilités formalisent ce que doivent être de telles combinaisons de comportements. Elles montrent clairement que faire de telles combinaisons impose nécessairement de fournir des connaissances préalables supplémentaires explicitant comment les différents comportements doivent être combinés entre eux.

Chapitre 7

Composition hiérarchique de descriptions

7.1 Objectifs et protocole expérimental

Dans cette expérience, nous cherchons à réécrire le comportement de phototaxie (Chapitre 4 section 4.1) en utilisant le modèle capteur pour `ThetaL` (Chapitre 5 section 5.2) au lieu du pré–traitement `Lum` (Chapitre 3 section 3.1). Nous notons *Cp_PhotoSL* les connaissances préalables associées à cette description.

7.2 Spécification

7.2.1.1 Choix des variables

Les variables retenues sont les deux variables caractérisant la source lumineuse `ThetaL`, `DistL`, ainsi que les variables correspondant aux huit capteurs de luminosité :`Lm0` à `Lm7` et, bien entendu, la variable motrice `Vrot`.

Variables pertinentes

- **`Vrot`** : domaine {-10, -9,…, +10}, $k_{\texttt{Vrot}} = 21$;
- **`ThetaL`** : domaine {-180, -170,…, +170}, $k_{\texttt{ThetaL}} = 36$;
- **`Lm0 à Lm7`** : domaine {0, 1,…, 511}, $k_{\texttt{Lmi}} = 512$.

7.2.1.2 Connaissances préalables de dépendance

La description du comportement porte sur la variable motrice `Vrot` et sur les variables sensorielles `ThetaL`, `Lm0` à `Lm7` et correspond à la distribution conjointe :

$$\mathrm{P}(\texttt{ThetaL Lm0..Lm7 Vrot} \mid Cp_PhotoSL), \tag{7.1}$$

que nous décomposons sous la forme du produit de deux distributions élémentaires :

$$\begin{aligned}&\mathrm{P}(\texttt{ThetaL Lm0..Lm7 Vrot} \mid Cp_PhotoSL)\\ &= \begin{vmatrix}\mathrm{P}(\texttt{ThetaL Lm0..Lm7} \mid Cp_PhotoSL)\\ \mathrm{P}(\texttt{Vrot} \mid \texttt{ThetaL Lm0..Lm7}\ Cp_PhotoSL).\end{vmatrix}\end{aligned} \tag{7.2}$$

L'objectif recherché est d'utiliser la description précédemment définie (Chapitre 5 section 5.2) comme un pré-traitement permettant d'obtenir la direction de la source lumineuse en fonction des valeurs retournées par les capteurs de luminosité. Ainsi, nous souhaitons déterminer la vitesse de rotation `Vrot` uniquement sur la base de l'information fournie par la variable `ThetaL`. Ce choix va se traduire par l'hypothèse d'indépendance conditionnelle suivante : connaissant la valeur de `ThetaL`, la vitesse de rotation `Vrot` est indépendante des valeurs retournées par les capteurs de luminosités `Lm0` à `Lm7`. Nous obtenons finalement la décomposition suivante :

$$\begin{aligned}&\mathrm{P}(\texttt{ThetaL Lm0..Lm7 Vrot} \mid Cp_PhotoSL)\\ &= \begin{vmatrix}\mathrm{P}(\texttt{ThetaL Lm0..Lm7} \mid Cp_PhotoSL)\\ \mathrm{P}(\texttt{Vrot} \mid \texttt{ThetaL}\ Cp_PhotoSL).\end{vmatrix}\end{aligned} \tag{7.3}$$

7.2.1.3 Connaissances préalables d'observation

Nous venons de décomposer la distribution conjointe en un produit de distributions élémentaires. Il reste à indiquer les valeurs des distributions élémentaires et la manière dont elles évoluent au vu de données expérimentales.

Tout comme dans la première expérience présentée au chapitre 4 section 4.1, nous choisissons de représenter la probabilité de `Vrot` conditionnée par `ThetaL` sous la forme d'un ensemble de lois normales paramétrées par la valeur de `ThetaL`. De plus, dans le cadre de cette expérience, les valeurs des paramètres des lois normales seront spécifiées a priori sous forme fonctionnelle, et ainsi ne seront pas identifiées sur la base d'un jeu de données expérimentales :

$$\begin{aligned}&\mathrm{P}(\texttt{Vrot} \mid [\texttt{ThetaL} = \texttt{thetaLi}]\ Cp_PhotoSL)\\ &= \mathbf{G}_{\mu_P(\texttt{thetaLi}),\ \sigma_P(\texttt{thetaLi})}(\texttt{Vrot}).\end{aligned} \tag{7.4}$$

Il reste à spécifier la distribution fixant la dépendance entre la variable `ThetaL` et les capteurs de luminosité `Lm0` à `Lm7`. Pour cela, nous allons simplement spécifier que cette distribution est obtenue sous la forme d'une question posée à la description définie plus haut (Chapitre 5 section 5.3) : *Cp_SL*.

$$\begin{aligned} &\mathrm{P}(\texttt{ThetaL Lm0..Lm7} \mid Cp_PhotoSL) \\ &= \mathrm{P}(\texttt{ThetaL Lm0..Lm7} \mid Cp_SL). \end{aligned} \tag{7.5}$$

7.3 Identification

Dans l'expérience du comportement phototaxie décrite au chapitre 4 section 4.1, nous avons spécifié « à la main » les paramètres des lois normales associées à la distribution sur `Vrot`, sous la forme d'un tableau. Ici, nous utilisons une forme fonctionnelle, traduisant le comportement recherché. Intuitivement, les espérances et écart-types doivent êtres choisis pour mener au comportement suivant :

- Plus l'orientation du robot est écartée de la source lumineuse plus sa vitesse de rotation doit être rapide et moins les contraintes sur la valeur exacte des vitesses sont importantes (i.e. plus l'écart-type est important). La vitesse est positive si la source est à sa gauche, négative sinon.
- Lorsque que le robot se trouve face à la source, `ThetaL` = 0, nous imposons au robot une vitesse de rotation nulle. La notion intuitive « d'imposer » se traduit par une plus forte contrainte sur les vitesses (écart type = 0,5).
- Lorsque le robot tourne le dos à la source lumineuse, nous voulons tourner du côté le plus éclairé (positivement pour `ThetaL` ≤ +170 et négativement pour `ThetaL` ≥ -180).

Nous avons spécifié les écart-types en fonction de la valeur de `ThetaL`. Lorsque le robot tourne le dos à la source lumineuse (la valeur absolue de `ThetaL` est grande) l'écart-type est relativement important (valeur 5,5). Plus le Khepera est orienté vers la source lumineuse et plus la valeur de l'écart-type diminue pour atteindre une valeur minimale de 0,5. La décroissance se fait selon une sigmoïde. La fonction obtenue est la suivante, représentée Figure 7-1 :

$$\begin{aligned} &\sigma_p(\texttt{ThetaL}) = 0{,}5 + 5{,}0 * \mathrm{sigmoid}_{\alpha,\beta}(|\texttt{ThetaL}|) \\ &\text{avec } \alpha = 90 \text{ et } \beta = 0{,}012. \end{aligned} \tag{7.6}$$

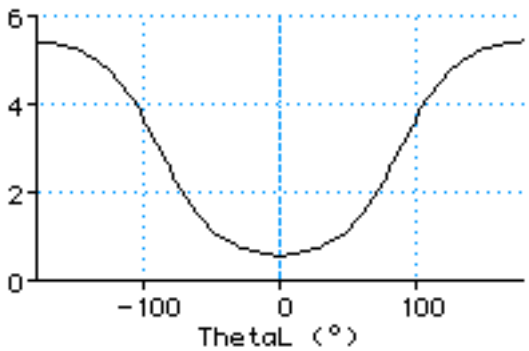


Figure 7-1 — La fonction $\sigma_p(\texttt{ThetaL})$.

La vitesse de rotation doit être négative pour une source lumineuse à gauche du Khepera (`ThetaL` < 0) et positive dans le cas contraire (`ThetaL` > 0). Plus le Khepera se trouvera face à la source lumineuse, plus la vitesse de rotation tendra vers zéro. En s'écartant de la

direction le menant vers la source lumineuse, la valeur absolue de la vitesse de rotation augmentera en suivant une fonction sigmoïde en `ThetaL`. La valeur maximale sera `Vrot` = -9, atteinte pour `ThetaL` = -180. Nous avons fixé le point d'inflexion de la sigmoïde à `ThetaL` = 0, la pente valant 0,005 en ce point. La fonction obtenue est la suivante, représentée Figure 7-2 :

$$\mu_p(\texttt{ThetaL}) = 18\left(\text{sigmoid}_{\alpha,\beta}(\texttt{ThetaL}) - 0{,}5\right) \quad \text{avec } \alpha = 0 \text{ et } \beta = 0{,}005. \tag{7.7}$$

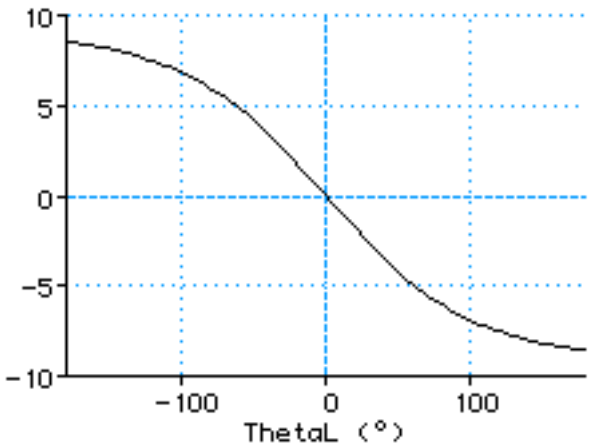


Figure 7-2 — La fonction sigmoïde $\mu_p(\texttt{ThetaL})$.

Par la suite nous nous réfèrerons à cette description par la connaissance préalable notée *Cp_PhotoSL* :

$$\begin{aligned} &P(\texttt{ThetaL Lm0..Lm7 Vrot} \mid Cp_PhotoSL) \\ &= P(\texttt{Vrot} \mid \texttt{ThetaL } Cp_PhotoSL)\,P(\texttt{ThetaL Lm0..Lm7} \mid Cp_PhotoSL). \end{aligned} \tag{7.8}$$

7.4 Utilisation

7.4.1 Programme et question probabiliste

Nous avons utilisé cette description pour piloter le Khepera de manière à ce qu'il se dirige vers une source lumineuse. Ce comportement réflexe est obtenu par la requête suivante :

1/ Descriptions utilisées :

Cp_PhotSL, Cp_SL

2/ Requête :

E_q : {Vrot}

E_c : {Lm0, …, Lm7}

E_i : {ThetaL}

Question : P(Vrot | Lm0 … Lm7 *Cp_PhotoSL*)

3/ Décision :

Tirage aléatoire

La question posée ici est :

$$\mathrm{P}(\mathtt{Vrot} \mid \mathtt{lm0...lm7}\ Cp_PhotoSL). \tag{7.9}$$

La réponse à cette question est donnée par :

$$\begin{aligned}
&\mathrm{P}(\mathtt{Vrot} \mid \mathtt{lm0...lm7}\ Cp_PhotoSL) \\
&= \frac{\mathrm{P}(\mathtt{Vrot\ lm0...lm7} \mid Cp_PhotoSL)}{\mathrm{P}(\mathtt{lm0...lm7} \mid Cp_PhotoSL)} \\
&= \frac{\sum_{\mathtt{ThetaL}} \mathrm{P}(\mathtt{Vrot} \mid \mathtt{ThetaL}\ Cp_PhotoSL)\mathrm{P}(\mathtt{ThetaL\ lm0...lm7} \mid Cp_SL)}{\sum_{\substack{\mathtt{Vrot}\\ \mathtt{ThetaL}}} \mathrm{P}(\mathtt{Vrot} \mid \mathtt{ThetaL}\ Cp_PhotoSL)\mathrm{P}(\mathtt{ThetaL\ lm0...lm7} \mid Cp_SL)}.
\end{aligned} \tag{7.10}$$

Le dénominateur de cette expression se simplifie en :

$$\begin{aligned}
&\sum_{\substack{\mathtt{Vrot}\\ \mathtt{ThetaL}}} \mathrm{P}(\mathtt{Vrot} \mid \mathtt{ThetaL}\ Cp_PhotoSL)\mathrm{P}(\mathtt{ThetaL\ lm0...lm7} \mid Cp_SL) \\
&= \sum_{\mathtt{ThetaL}} \left(\mathrm{P}(\mathtt{ThetaL\ lm0...lm7} \mid Cp_SL) \left(\sum_{\mathtt{Vrot}} \mathrm{P}(\mathtt{Vrot} \mid \mathtt{ThetaL}\ Cp_PhotoSL) \right) \right) \\
&= \mathrm{P}(\mathtt{lm0...lm7} \mid Cp_SL).
\end{aligned} \tag{7.11}$$

En remplaçant le dénominateur dans (7.10) nous obtenons :

$$\begin{aligned} &P(\texttt{Vrot} \mid \texttt{lm0...lm7}\ Cp_PhotoSL) \\ &= \frac{\sum_{\texttt{ThetaL}} P(\texttt{Vrot} \mid \texttt{ThetaL}\ Cp_PhotoSL) P(\texttt{ThetaL lm0...lm7} \mid Cp_SL)}{P(\texttt{lm0...lm7} \mid Cp_SL)} \\ &= \sum_{\texttt{ThetaL}} \left(P(\texttt{Vrot} \mid \texttt{ThetaL}\ Cp_PhotoSL) \frac{P(\texttt{ThetaL lm0...lm7} \mid Cp_SL)}{P(\texttt{lm0...lm7} \mid Cp_SL)} \right). \end{aligned} \tag{7.12}$$

Puis, finalement, en appliquant la règle du produit, le question se ramène à :

$$\begin{aligned} &P(\texttt{Vrot} \mid \texttt{lm0...lm7}\ Cp_PhotoSL) \\ &= \sum_{\texttt{ThetaL}} \left(P(\texttt{Vrot} \mid \texttt{ThetaL}\ Cp_PhotoSL) P(\texttt{ThetaL} \mid \texttt{lm0...lm7}\ Cp_SL) \right). \end{aligned} \tag{7.13}$$

La distribution P(ThetaL | lm0…lm7 *Cp_PhotoSL*) est une requête posée à la description *Cp_SL* que nous pouvons détailler ainsi :

1/ Descriptions utilisées :

Cp_SL

2/ Requête :

E_q : {ThetaL}

E_c : {Lm0, …, Lm7}

E_i : {DistL}

Question : P(ThetaL | Lm0 … Lm7 *Cp_SL*)

Ici il n'y a pas de décision explicitement fixée puisque la distribution résultant de la requête sera utilisée dans son intégralité au sein de l'expression (7.10).

Cette distribution est obtenue par marginalisation sur DistL :

$$\begin{aligned} &P(\texttt{ThetaL} \mid \texttt{lm0...lm7}\ Cp_SL) \\ &= \frac{1}{Z} \sum_{\texttt{DistL}} \prod_{i=0}^{7} P(\texttt{lmi} \mid \texttt{ThetaL DistL}\ Cp_li). \end{aligned} \tag{7.14}$$

L'expression finale de notre requête est :

$$\begin{aligned} &P(\texttt{Vrot} \mid \texttt{lm0...lm7}\ Cp_PhotoSL) \\ &= \frac{1}{Z} \sum_{\texttt{ThetaL}} \left(P(\texttt{Vrot} \mid \texttt{ThetaL}\ Cp_PhotoSL) \sum_{\texttt{DistL}} \left(\prod_{i=0}^{7} P(\texttt{lmi} \mid \texttt{ThetaL DistL}\ Cp_li) \right) \right). \end{aligned} \tag{7.15}$$

Nous voyons ici que pour répondre à la requête initiale, nous avons généré une requête intermédiaire à la description *Cp_SL*. Le résultat de cette requête se traduit par une distribution de probabilité sur la variable ThetaL. Ainsi l'incertitude sur cette variable est

propagée. Le coût de cette propagation apparaît dans (7.15) sous la forme d'une somme sur la variable `ThetaL` (qui correspond à la marginalisation sur cette variable).

Pour des raisons d'optimisation, il est possible alors de minimiser ce coût en faisant une décision sur la valeur de la variable `ThetaL`. Ceci se ramène à poser successivement deux requêtes, la première permettant de déterminer la valeur de la variable `ThetaL` connaissant les valeurs des capteurs de luminosité `Lm0` à `Lm7` selon la description *CP_SL* puis, une fois cette valeur connue, décider de la valeur de `Vrot` connaissant la valeur de `ThetaL` selon la description *Cp_PhotoSL* :

La première requête est :

1/ Descriptions utilisées :

Cp_SL

2/ Requête :

E_q : {ThetaL}

E_c: {Lm0, …, Lm7}

E_i: {DistL}

Question : P(ThetaL | Lm0 … Lm7 *Cp_PhotoSL*)

La réponse à cette première requête consiste à tirer au hasard selon la distribution sur `ThetaL` calculée selon (7.14).

La deuxième requête consiste à chercher `Vrot` en ne considérant connue que la variable `ThetaL` :

1/ Descriptions utilisées :

Cp_PhotoSL, Cp_SL

2/ Requête :

E_q : {Vrot}

E_c: {ThetaL }

E_i: {Lm0, …, Lm7}

Question : P(Vrot | ThetaL *Cp_PhotoSL*)

3/ Décision :

Tirage aléatoire

Cette fois ci, la réponse est obtenue directement par la distribution élémentaire définie dans les connaissances préalables associées à la description *Cp_PhotoSL*, que nous rappelons :

$$\mathrm{P}\left(\texttt{Vrot} \mid [\texttt{ThetaL} = \texttt{thetaLi}]\ \mathit{Cp_PhotoSL}\right) = \mathbf{G}_{\mu_P(\texttt{thetaLi}),\ \sigma_P(\texttt{thetaLi})}(\texttt{Vrot}). \qquad (7.16)$$

7.4.2 Résultat

Les figures suivantes présentent les résultats obtenus en propageant l'incertitude sur `ThetaL`. Pour chaque figure, la distribution à gauche est le résultat de la question posée à la description du modèle capteur *Cp_SL*, P(`Theta` | `Lm0...Lm7` *Cp_SL*) ; la distribution de droite est le résultat global obtenu pour la vitesse de rotation `Vrot`, P(`Vrot` | `Lm0...Lm7` *Cp_PhotoSL*).

Les deux premières figures correspondent à une situation où le robot est en présence d'une seule source lumineuse située à l'avant gauche du robot (`Theta` = -10°). Dans la première situation (Figure 7-3), le modèle capteur trouve la direction de la source sans incertitude (la probabilité de `ThetaL` = -10° vaut 1), la distribution sur `Vrot` est centrée sur –2 avec une faible incertitude. Dans le cas suivant (Figure 7-4), nous avons éloigné la source lumineuse, ce qui a pour effet d'accroître l'incertitude sur l'orientation de la source. Cette incertitude se répercute sur la vitesse de rotation, la vitesse de rotation –2 est toujours la plus probable, mais la réponse est plus nuancée et autorise des vitesses de rotation entre –5 et +1.

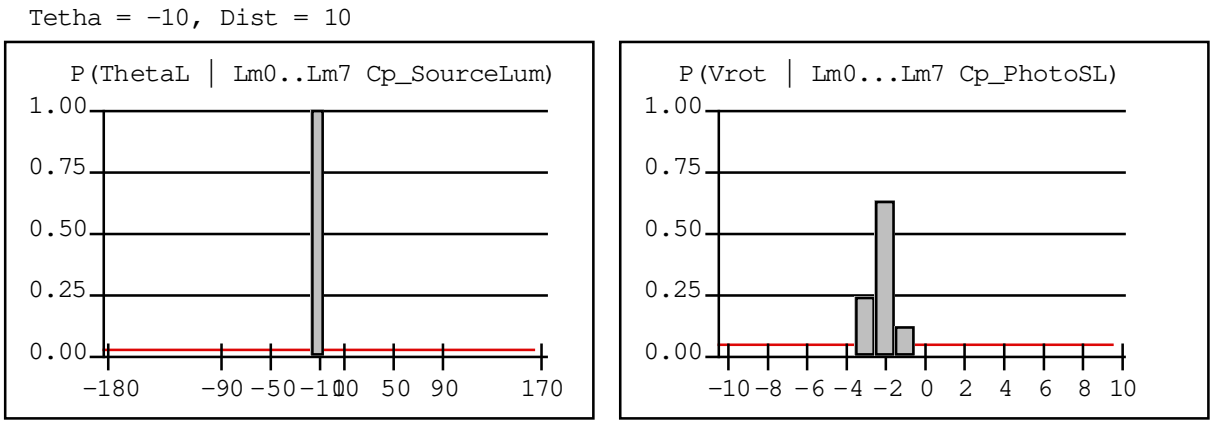


Figure 7-3 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle.
La source lumineuse est proche du robot dans le secteur avant gauche (`Theta` = –10°, `Dist` = 10).

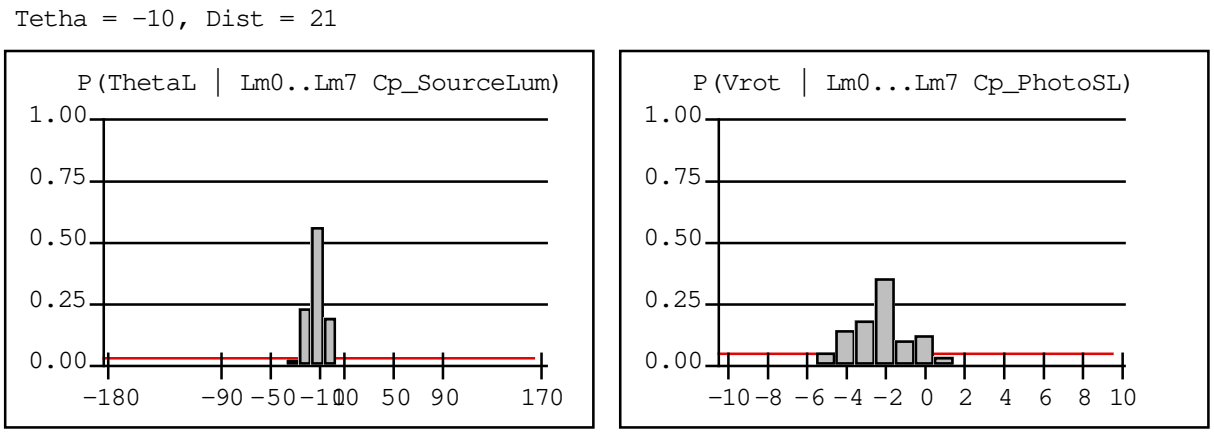


Figure 7-4 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle.
La source lumineuse est relativement éloignée du robot dans le secteur avant gauche (`Theta` = –10°, `Dist` = 21).

Pour les deux dernières figures, le robot est en présence de deux sources lumineuses, une première à droite du robot (90°) et la seconde à sa gauche (-90°). Lorsque ces deux sources se trouvent à égale distance du robot (`Dist` = 20) (Figure 7-5), le modèle capteur indique la présence des deux sources avec une probabilité équivalente. La commande en rotation restitue la connaissance des deux sources par deux alternatives équiprobables, soit tourner fortement sur la gauche, soit tourner fortement sur la droite. Toutefois, dès qu'une source lumineuse se trouve plus proche du robot, une seule des deux alternatives est obtenue, se diriger vers celle-ci. Comme le montre la Figure 7-6, ceci provient du modèle capteur, pour lequel plus une source est proche et plus il sait l'identifier avec certitude. Ainsi, le résultat obtenu est un comportement consistant à se diriger vers la source la plus proche du robot.

```
Source1 : Tetha = 90, Dist = 20
Source2 : Tetha = −90, Dist = 20
```

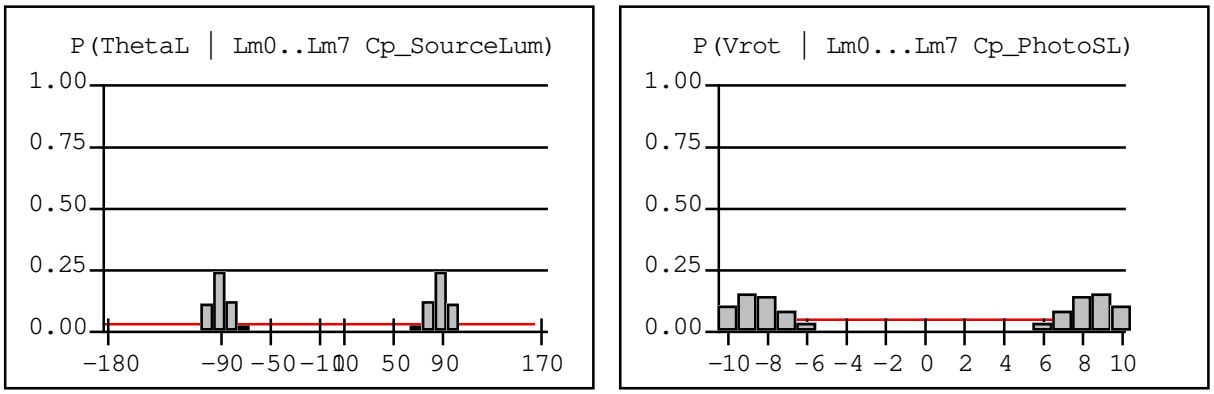


Figure 7-5 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle.
Deux sources lumineuses à égales distances du robot, l'une à sa droite (+90°) l'autre à sa gauche (-90°).

```
Source1 : Tetha = +90, Dist = 20
Source2 : Tetha = −90, Dist = 17
```

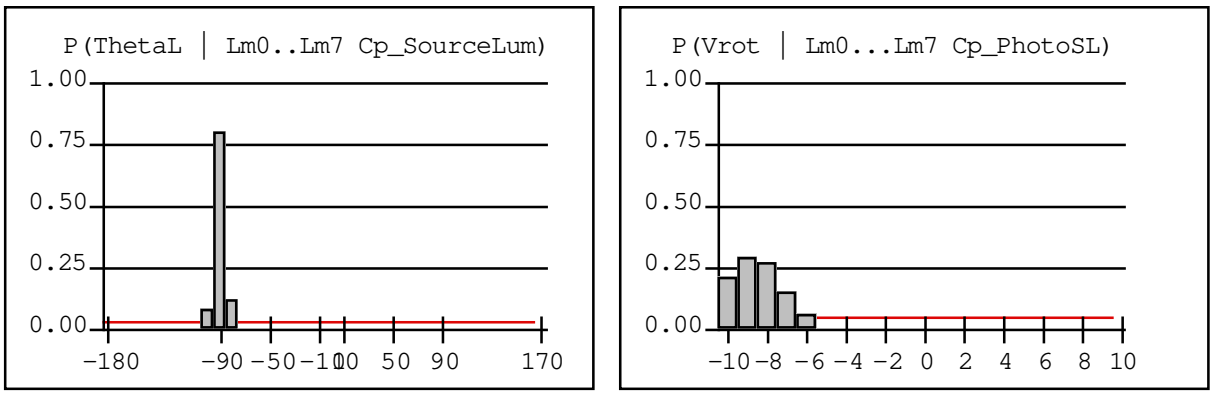


Figure 7-6 — Composition hiérarchique d'un modèle capteur et d'une loi de contrôle.
Deux sources lumineuses, l'une à la droite du Khepera (+90°) l'autre à sa gauche (-90°), la source de gauche est plus proche du robot.

7.5 Enseignements et discussion

Alors que dans le chapitre précédent nous avons vu que les comportements (descriptions) peuvent être combinés entre eux pour produire des comportements plus complexes, nous voyons dans cette expérience que les descriptions peuvent être « hiérarchisées » pour élaborer de manière incrémentale des descriptions de plus en plus abstraites.

Une description de niveau *n* est ainsi utilisée pour inférer une variable utilisée par une description de niveau *n*+1. Dans l'expérience décrite ici, par exemple, la description `Cp_SL` permet d'inférer `ThetaL` qui est utilisé par la description *Cp_PhotoSL*. Comme nous le verrons de nouveau au Chapitre 10 à la section 10.4, la variable inférée par le niveau n peut soit faire l'objet d'une décision et être passée comme une valeur d'entré au niveau *n*+1, soit être passée, sans perte d'information, en tant que distribution de probabilité.

Chapitre 8

Reconnaissance de situations

Les deux expériences présentées dans cette section illustrent l'application de notre méthode dans le cadre d'une tâche de reconnaissance. Dans la première expérience, nous cherchons à fournir au Khepera le moyen de détecter lorsqu'il se trouve à sa base. La deuxième expérience est une tâche de reconnaissance d'objets dans laquelle le Khepera sera en mesure de reconnaître des objets déjà rencontrés et de détecter lorsqu'il se trouve en présence d'un nouvel objet.

8.1 Reconnaissance de la base

8.1.1 Objectifs et protocole expérimental

L'objectif de cette expérience est de construire une description permettant au Khepera de reconnaître lorsqu'il se trouve en un endroit particulier de la scène : sa base.

8.1.2 Spécification

8.1.2.1 Choix des variables

Pour cette tâche de reconnaissance de situation, nous avons choisi de travailler avec les variables sensorielles de base associées aux capteurs de proximités et de luminosités : `Px0` à `Px7` (chacune des variables `Pxi` varie de 0 à 1023 soit 1024 cas possibles) et `Lm0` à `Lm7` (chacune des variables `Lmi` varie de 0 à 511 soit 512 cas possibles). Ce choix traduit l'hypothèse qu'il est possible de reconnaître la situation « être à la base » au vu simplement des valeurs retournées par ces variables. Il reste encore à spécifier une dernière variable :

`Base`, cette dernière pouvant prendre deux valeurs {0, 1}. [`Base`=1] lorsque le robot est la base, et [`Base`=0] dans le cas contraire.

Variables pertinentes

- **`Base`** : domaine {0,1}, $k_{\texttt{Base}} = 2$;
- **`Lm0 à Lm7`** : domaine {0, 1,…, 511}, $k_{\texttt{Lmi}} = 512$.
- **`Px0 à Px7`** : domaine {0, 1,…, 1023}, $k_{\texttt{Pxi}} = 1024$.

8.1.2.2 Connaissances préalables de dépendance

L'expression des connaissances préalables de dépendance va prendre une dimension particulièrement importante dans le cas présent. Le choix des variables retenues permet d'exprimer la description comme la probabilité conjointe suivante :

$$\mathrm{P}(\texttt{Base Lm0...Lm7 Px0...Px7} \mid D\ C). \tag{8.1}$$

Le nombre de cas possibles est gigantesque : $1024^8 \times 512^8 \times 2 \sim 10^{46}$ cas. Encore une fois, par l'expression d'hypothèses d'indépendances conditionnelle nous allons être en mesure de réduire considérablement la dimension pour représenter cette distribution. En effet, conditionnellement à la connaissance du fait que le robot se trouve ou ne se trouve pas à sa base, nous allons considérer que les valeurs des différentes variables sensorielles sont indépendantes les unes des autres. La Figure 8-1 illustre ce choix :

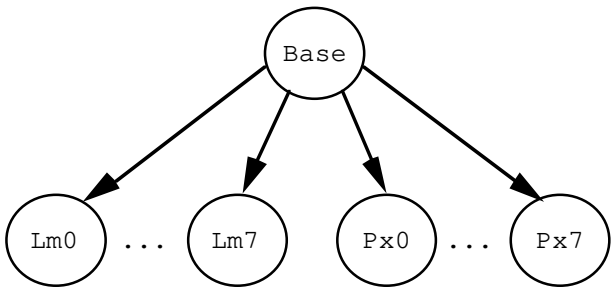


Figure 8-1 — Relation de dépendances entre les 16 variables sensorielles et la variables `Base`. Cette fait apparaître les indépendances conditionnelles entre variables sensorielles connaissant la variable `Base`.

Finalement, nous exprimons la distribution conjointe sous la forme de la décomposition suivante :

$$\mathrm{P}(\texttt{Base Lm0...Lm7 Px0...Px7} \mid D\ C) = \left| \begin{array}{l} \mathrm{P}(\texttt{Base} \mid D\ C) \\ \mathrm{P}(\texttt{Lm0} \mid \texttt{Base}\ D\ C) \\ \ldots \\ \mathrm{P}(\texttt{Lm7} \mid \texttt{Base}\ D\ C) \\ \mathrm{P}(\texttt{Px0} \mid \texttt{Base}\ D\ C) \\ \ldots \\ \mathrm{P}(\texttt{Px7} \mid \texttt{Base}\ D\ C). \end{array} \right. \tag{8.2}$$

Cette forme de décomposition n'est pas sans rappeler celle obtenue dans l'expression de la description du modèle capteur de la source de luminosité (Chapitre 5). D'une certaine façon,

nous pouvons voir cette reconnaissance de situation comme la spécification d'un modèle capteur particulier permettant de détecter la base.

8.1.2.3 Connaissances préalables d'observation

Ne sachant rien d'autre, soit le robot est à la base soit il n'y est pas. Nous ne désirons pas spécifier plus de connaissance et ainsi nous associons une distribution uniforme à la probabilité a priori d'être à la base :

$$\mathrm{P}\left([\texttt{Base}=1] \mid D\ C\right) = \mathrm{P}\left([\texttt{Base}=0] \mid D\ C\right) = \mathbf{U} = \frac{1}{2}. \tag{8.3}$$

Pour fixer les formes paramétriques des distributions élémentaires associées à chaque variable sensorielle, nous allons distinguer le cas où le robot se trouve à la base et le cas contraire. Lorsque le robot ne se trouve pas à la base, nous considérons que le robot peut se trouver dans une situation quelconque. Nous traduisons « situation quelconque » tout simplement en spécifiant que les capteurs sensoriels peuvent prendre une valeur quelconque (distribution uniforme). L'identification du cas où le robot se trouve à la base va consister à placer effectivement le robot à sa base et à enregistrer les valeurs lues sur ses capteurs. De cet ensemble de valeurs, nous ne retiendrons que l'espérance empirique du premier et du second moment. Ce choix d'observables correspond à représenter les distributions des valeurs des variables sensorielles dans le cas où le robot est à la base sous la forme de loi normales :

Robot hors de la base :

$$\begin{aligned}
&\mathrm{P}\left(\texttt{Lmi} \mid [\texttt{Base}=0]\ D\ C\right) = \mathrm{P}\left(\texttt{Lmi} \mid [\texttt{Base}=0]\ C\right) = \mathrm{U} = \frac{1}{512},\\
&\mathrm{P}\left(\texttt{Pxi} \mid [\texttt{Base}=0]\ D\ C\right) = \mathrm{P}\left(\texttt{Pxi} \mid [\texttt{Base}=0]\ C\right) = \mathrm{U} = \frac{1}{1024}\ ;
\end{aligned} \tag{8.4}$$

Robot à la base :

$$\begin{aligned}
&\mathrm{P}\left(\texttt{Lmi} \mid [\texttt{Base}=1]\ D\ C\right) = \mathbf{G}_{\mu li,\ \sigma li}(\texttt{Lmi}),\\
&\mathrm{P}\left(\texttt{Pxi} \mid [\texttt{Base}=1]\ D\ C\right) = \mathbf{G}_{\mu pi,\ \sigma pi}(\texttt{Pxi}).
\end{aligned}$$

8.1.3 Identification

La phase d'identification est tres simple. Nous situons le Khepera à sa base et nous enregistrons les données pendant 30 secondes environ (soit 300 données échantillonnées à une cadence de 10Hz). Au cours de l'enregistrement, nous forçons le Khepera à effectuer quelques mouvements de rotation sur place de manière à éviter d'identifier une situation trop spécifique.

Par la suite nous nous réfèrerons à cette description par les connaissances préalables notées *Cp_DétectBase* (associée implicitement au jeu de données utilisé pour l'identification) :

$$\begin{aligned} & P(\texttt{Base Lm0..Lm7 Px0...Px7} \mid \textit{Cp_DétectBase}) \\ & = \left| \begin{array}{l} P(\texttt{Base} \mid \textit{Cp_DétectBase}) \\ P(\texttt{Lm0} \mid \texttt{Base } \textit{Cp_DétectBase}) \\ \ldots \\ P(\texttt{Lm7} \mid \texttt{Base } \textit{Cp_DétectBase}) \\ P(\texttt{Px0} \mid \texttt{Base } \textit{Cp_DétectBase}) \\ \ldots \\ P(\texttt{Px7} \mid \texttt{Base } \textit{Cp_DétectBase}). \end{array} \right. \end{aligned} \tag{8.5}$$

8.1.4 Utilisation

8.1.4.1 Programme et question probabiliste

L'utilisation de cette description va consister à déterminer la probabilité que la variable `Base` prenne la valeur 1, connaissant les valeurs des différentes variables sensorielles. Cela correspond à la requête suivante :

1/ Descriptions utilisées :

Cp_DétectBase

2/ Requête :

E_q : {Base}

E_c : {Lm0, …, Lm7, Px0, …, Px7}

E_i : Ø

Question : P(Base | Lm0 … Lm7 Px0 … Px7 *Cp_DétecBase*)

3/ Décision :

Tirage aléatoire

Nous notons en minuscules les variables dont les valeurs sont connues (`lm0`, …, `lm7`, `px0`,…, `px7`). Le calcul de la question posée se ramène à l'expression suivante :

$$\begin{aligned} & P(\texttt{Base} \mid \texttt{lm0...lm7 px0...px7 } \textit{Cp_DétectBase}) \\ & = \frac{P(\texttt{Base lm0...lm7 px0...px7} \mid \textit{Cp_DétectBase})}{P(\texttt{lm0...lm7 px0...px7} \mid \textit{Cp_DétectBase})} \\ & = \frac{P(\texttt{Base lm0...lm7 px0...px7} \mid \textit{Cp_DétectBase})}{\sum_{\texttt{Base}} P(\texttt{Base lm0...lm7 px0...px7} \mid \textit{Cp_DétectBase})}. \end{aligned} \tag{8.6}$$

Le numérateur de cette expression est directement calculable à partir de la décomposition fixée dans les connaissances préalables. Connaissant les valeurs des différentes variables sensorielles, nous sommes en mesure de calculer leurs probabilités selon la valeur que prend

la variable `Base`. Nous notons P1 la valeur du numérateur obtenue pour [`Base` = 1] et P0 la valeur obtenue pour [`Base` = 0]. Ainsi, l'expression de la probabilité d'être à la base connaissant les valeurs des variables sensorielles est :

$$\mathrm{P}\left([\texttt{Base}=1] \mid \texttt{lm0...lm7 px0...px7}\ Cp_DétectBase\right) = \frac{P1}{P1+P0}. \tag{8.7}$$

Il reste à calculer la valeur de P1 et de P0 :

$$\begin{aligned} P1 &= \mathrm{P}\left([\texttt{Base}=1]\ \texttt{lm0...lm7 px0...px7} \mid Cp_DétectBase\right) \\ &= \begin{vmatrix} \mathrm{P}\left([\texttt{Base}=1] \mid Cp_DétectBase\right) \\ \mathrm{P}\left(\texttt{lm0} \mid [\texttt{Base}=1]\ Cp_DétectBase\right) \\ \texttt{K} \\ \mathrm{P}\left(\texttt{px7} \mid [\texttt{Base}=1]\ Cp_DétectBase\right) \end{vmatrix} \\ &= \frac{1}{2}\prod_{i=0}^{7}\mathbf{G}_{\mu li,\ \sigma li}(\texttt{lmi})\prod_{j=0}^{7}\mathbf{G}_{\mu pj,\ \sigma pj}(\texttt{px}j). \end{aligned} \tag{8.8}$$

$$\begin{aligned} P0 &= \mathrm{P}\left([\texttt{Base}=0]\ \texttt{lm0...lm7 px0...px7} \mid Cp_DétectBase\right) \\ &= \begin{vmatrix} \mathrm{P}\left([\texttt{Base}=0] \mid Cp_DétectBase\right) \\ \mathrm{P}\left(\texttt{lm0} \mid [\texttt{Base}=0]\ Cp_DétectBase\right) \\ \texttt{K} \\ \mathrm{P}\left(\texttt{px7} \mid [\texttt{Base}=0]\ Cp_DétectBase\right) \end{vmatrix} \\ &= \frac{1}{2}\left(\frac{1}{512}\right)^{8}\left(\frac{1}{1024}\right)^{8}. \end{aligned} \tag{8.9}$$

8.1.4.2 Résultat

Après une trentaine de secondes d'apprentissage, nous constatons que le robot est capable de reconnaître sa base au bout de quelques secondes d'hésitation. Cette hésitation est due en partie au cours temps d'apprentissage et en partie au fait que l'apprentissage s'est fait en déplaçant le robot avec la main. Ce que le Khepera a appris est donc « être à la base avec une main au-dessus de moi » ce qui est sensiblement différent de la situation « être à la base », notamment parce que la main perturbe les capteurs de lumières avec son ombre. La même expérience refaite en montrant au robot ce qu'est sa base à l'aide du manche à balai permet d'obtenir de meilleurs résultats.

Figure 8-2 — Retour à la base.
Le Khepera rentre à sa base en suivant un comportement de phototaxie combiné avec le comportement évitement d'obstacle. La détection de la base est obtenue par une phase d'apprentissage durant laquelle l'opérateur indique les différentes situations sensorielles correspondantes à cette situation.
(film : Film—Chp8.mov ; durée : 1 minute 45 secondes).

Ce quatrième film illustre l'utilisation de la détection de la base. On voit tout d'abord le Khepera rentrer à sa base grâce au comportement obtenu par composition de l'évitement d'obstacle et de la phototaxie (présentée au Chapitre 6 section 6.2). Le Khepera qui n'a pas appris à reconnaître sa base se met alors à tourner sur place sous la lumière. On voit ensuite un apprentissage de ce qu'est la base par la présentation au robot de différentes situations sensorielles correspondant à la base. On voit, enfin, un nouveau retour à la base, suivi, après quelques hésitations, par une reconnaissance de la base et par l'immobilisation du robot.

8.2 Reconnaissance d'objets

8.2.1 Objectifs et protocole expérimental

Les objectifs de cette expérience sont très similaires à ceux de l'expérience précédente. Nous souhaitons fournir au Khepera la capacité de reconnaître des objets. Cette fois ci, ce n'est plus une situation que l'on cherche à détecter mais plusieurs ; chaque objet est une situation particulière. De plus, nous désirons que le Khepera soit en mesure de détecter la nouveauté. Lorsque nous présenterons un objet que le Khepera n'a pas eu l'occasion de rencontrer préalablement, nous voulons qu'il soit en mesure de détecter qu'il s'agit d'un nouvel objet et ainsi de créer un nouveau modèle pour décrire cet objet.

Nous illustrons ici comment nous pouvons utiliser un comportement réactif simple pour acquérir de nouvelles informations sur l'environnement. Nous avons ainsi construit le comportement de reconnaissance d'objets à partir du comportement réactif de suivi de contour.

En appliquant le comportement de suivi, le Khepera fait le tour de l'objet qu'il a à reconnaître. Ce faisant il calcule quatre nouvelles variables qui vont lui permettre de classifier

les objets : le nombre de virages à gauche, le nombre de virages à droite, le périmètre et la longueur de la plus grande ligne droite.

En utilisant l'inférence bayésienne et des descriptions probabilistes des objets acquises par l'expérience, le Khepera est ainsi capable de distinguer, les uns des autres, toute une collection d'objets. Qui plus est, en utilisant une description de référence uniforme, il est en mesure de détecter lorsqu'il se trouve en présence d'un nouvel objet.

8.2.2 Spécification

8.2.2.1 Choix des variables

Le Khepera ne dispose pas de capteurs lui permettant de caractériser directement un objet. Toutefois, nous disposons d'un comportement de suivi de contour. Grâce à ce comportement, le Khepera est capable d'effectuer le tour d'un objet. Sur la base des informations proprioceptives du robot (nombre de rotations des moteurs associées aux roues gauche et droite), une fois le tour achevé, nous pouvons calculer différentes variables caractérisant l'objet comme le nombre de virage à droite, le nombre de virage à gauche, le périmètre de l'objet… Nous utilisons le suivi de contour comme protocole de mesure, identifiant ainsi différentes variables caractéristiques d'un objet.

Nous représentons le contour d'un objet comme une suite de segments (translation) effectués entre deux virages droits ou gauches du Khepera. Nous considérons que le Khepera aura effectué un virage lorsque sa rotation depuis le dernier virage sera supérieur à un angle fixé à l'avance (ici 45°). La mesure de rotation du robot est obtenue sur la base des informations fournies par les capteurs des positions des roues (`PosG`, `PosD`). Entre deux virages, nous mémorisons la valeur de translation calculée en se basant également sur les capteurs de positions de roues.

Sur la base de ces informations nous avons choisi de ne retenir que quatre variables qui sont :

`Dnv` : l'écart entre le nombre de virage à gauche et le nombre de virage à droite ;

`Mnv` : le plus grand nombre de virages effectués soit sur la droite soit sur la gauche ;

`Mld` : la plus grande translation parcourue ;

`Per` : le périmètre, la translation totale parcourue.

Nous utilisons, de plus, une variable supplémentaire afin de différencier les objets reconnus.

`Obj_id` : indice pour identifier les objets. Cet indice peut varier de 0 à 15. Nous réservons l'indice 0 pour l'objet inconnu grâce auquel nous détecterons la nouveauté.

Variables pertinentes

- **`Dnv`** : domaine {0 , 1, …, 49}, $k_{\texttt{Dnv}} = 50$;
- **`Mnv`** : domaine {0 , 1, …, 99}, $k_{\texttt{Mnv}} = 100$;
- **`Mld`** : domaine {0 , 1, …, 999}, $k_{\texttt{Mld}} = 1000$;
- **`Per`** : domaine {0 , 1, …, 9999}, $k_{\texttt{Per}} = 10000$;
- **`Obj_id`** : domaine {0, 1,…, 15}, $k_{\texttt{Obj_id}} = 16$.

8.2.2.2 Connaissances préalables de dépendance

La description porte sur les variables `Obj_id`, `Per`, `Mld`, `Mnv`, `Dnv` et est donc associée la distribution conjointe :

$$\mathrm{P}\left(\texttt{Obj_id Dnv Mnv Mld Per} \mid D\ C\right). \tag{8.10}$$

Nous décomposons cette expression en appliquant la règle [R1] de la manière suivante :

$$\begin{aligned} &\mathrm{P}\left(\texttt{Obj_id Dnv Mnv Mld Per} \mid D\ C\right) \\ &= \begin{vmatrix} \mathrm{P}\left(\texttt{Obj_id} \mid D\ C\right) \\ \mathrm{P}\left(\texttt{Dnv} \mid \texttt{Obj_id}\ D\ C\right) \\ \mathrm{P}\left(\texttt{Mnv} \mid \texttt{Dnv Obj_id}\ D\ C\right) \\ \mathrm{P}\left(\texttt{Mld} \mid \texttt{Mnv Dnv Obj_id}\ D\ C\right) \\ \mathrm{P}\left(\texttt{Per} \mid \texttt{Mld Mnv Dnv Obj_id}\ D\ C\right). \end{vmatrix} \end{aligned} \tag{8.11}$$

puis, à nouveau, nous exprimons des hypothèses d'indépendance conditionnelle qui consistent à considérer les variables caractérisant les objets (`Per`, `Mld`, `Mnv`, `Dnv`) indépendantes les unes des autres, conditionnellement à la connaissance de l'objet au quels elles réfèrent :

$$\begin{aligned} &\mathrm{P}\left(\texttt{Obj_id Dnv Mnv Mld Per} \mid D\ C\right) \\ &= \begin{vmatrix} \mathrm{P}\left(\texttt{Obj_id} \mid D\ C\right) \\ \mathrm{P}\left(\texttt{Dnv} \mid \texttt{Obj_id}\ D\ C\right) \\ \mathrm{P}\left(\texttt{Mnv} \mid \texttt{Obj_id}\ D\ C\right) \\ \mathrm{P}\left(\texttt{Mld} \mid \texttt{Obj_id}\ D\ C\right) \\ \mathrm{P}\left(\texttt{Per} \mid \texttt{Obj_id}\ D\ C\right). \end{vmatrix} \end{aligned} \tag{8.12}$$

8.2.2.3 Connaissances préalables d'observation

A priori nous pouvons demander au Khepera de reconnaître n'importe quel objet, qu'il soit nouveau ou bien qu'il est déjà été identifié. Nous notons $\mathrm{N}_{\texttt{obj_c}}$ le nombre d'objets que le robot connaît à un instant donné.

$$\mathrm{P}\left(\texttt{Obj_id} \mid D\ C\right) = \mathbf{U} = \frac{1}{N_{\texttt{Obj_c}}}. \tag{8.13}$$

Au fur et à mesure des apprentissages, cette distribution évoluera avec le nombre d'objets connus.

Pour les distributions suivantes, nous distinguons le cas de l'objet inconnu ([Obj_id = 0]) des autres cas. Nous modélisons l'objet inconnu en indiquant pour chacune des variables une loi uniforme sur son domaine de valeur :

$$
\begin{aligned}
\mathrm{P}\left(\mathtt{Dnv} \,\middle|\, \left[\mathtt{Obj_id}=0\right] D\ C\right) &= \mathbf{U} = \frac{1}{k_{\mathtt{Dnv}}},\\
\mathrm{P}\left(\mathtt{Mnv} \,\middle|\, \left[\mathtt{Obj_id}=0\right] D\ C\right) &= \mathbf{U} = \frac{1}{k_{\mathtt{Mnv}}},\\
\mathrm{P}\left(\mathtt{Mld} \,\middle|\, \left[\mathtt{Obj_id}=0\right] D\ C\right) &= \mathbf{U} = \frac{1}{k_{\mathtt{Mld}}},\\
\mathrm{P}\left(\mathtt{Per} \,\middle|\, \left[\mathtt{Obj_id}=0\right] D\ C\right) &= \mathbf{U} = \frac{1}{k_{\mathtt{Per}}}.
\end{aligned}
\tag{8.14}
$$

Pour les objets que nous cherchons à apprendre, nous choisissons de représenter les distributions associées aux variables `Dnv` et `Mnv` par des lois de succession de Laplace paramétrées par l'indice de chacun des objets.

$$
\begin{aligned}
&\text{pour chaque objet d' indice i} \neq 0\\
&\mathrm{P}\left(\mathtt{Dnv} \,\middle|\, \left[\mathtt{Obj_id}=\mathtt{i}\right] D\ C\right) = \mathbf{L}(\mathtt{Dnv})\\
\mathrm{P}\left(\left[\mathtt{Dnv}=\mathtt{dnv}_{\mathtt{j}}\right] \,\middle|\, \left[\mathtt{Obj_id}=\mathtt{i}\right] D\ C\right) &= \frac{n_{\mathtt{dnv}j}[i]+1}{N[i]+k_{\mathtt{Dnv}}},
\end{aligned}
\tag{8.15}
$$

où $n_{\mathtt{dnv}j}$ [i] est le nombre de fois où, pour l'objet d'indice i, la valeur $\mathtt{dnv}_{\mathtt{j}}$ de la variable `Dnv` a été observée, N[i] le nombre d'observations de cet objet et $k_{\mathtt{Dnv}}$ le nombre de cas possibles pour la variable `Dnv`.

$$
\begin{aligned}
&\text{pour chaque objet d' indice i} \neq 0\\
&\mathrm{P}\left(\mathtt{Mnv} \,\middle|\, \left[\mathtt{Obj_id}=\mathtt{i}\right] D\ C\right) = \mathbf{L}(\mathtt{Mnv})\\
\mathrm{P}\left(\left[\mathtt{Mnv}=\mathtt{mnv}_{\mathtt{j}}\right] \,\middle|\, \left[\mathtt{Obj_id}=\mathtt{i}\right] D\ C\right) &= \frac{n_{\mathtt{mnv}j}[i]+1}{N[i]+k_{\mathtt{Mnv}}}.
\end{aligned}
\tag{8.16}
$$

Enfin, nous choisissons de représenter les distributions associées aux variables `Mld` et `Per` par des lois de Gauss paramétrées par l'indice de chacun des objets identifiés.

$$
\begin{aligned}
&\text{pour chaque objet d' indice i} \neq 0\\
&\mathrm{P}\left(\mathtt{Mld} \,\middle|\, \left[\mathtt{Obj_id}=\mathtt{i}\right] D\ C\right) = \mathbf{G}_{\mu_{\mathtt{Mld}}[\mathtt{i}],\sigma_{\mathtt{Mld}}[\mathtt{i}]}(\mathtt{Mld})\\
&\mathrm{P}\left(\mathtt{Per} \,\middle|\, \left[\mathtt{Obj_id}=\mathtt{i}\right] D\ C\right) = \mathbf{G}_{\mu_{\mathtt{Per}}[\mathtt{i}],\sigma_{\mathtt{Per}}[\mathtt{i}]}(\mathtt{Per}).
\end{aligned}
\tag{8.17}
$$

8.2.3 Identification

L'identification consiste à mettre à jour les paramètres des différentes lois précédentes. Pour cela, nous présentons l'objet au robot, il effectue le tour de cet objet en appliquant le comportement de suivi de contour. Une fois le tour accompli, nous disposons des valeurs des variables décrivant l'objet. Deux cas peuvent se présenter, soit nous sommes en présence d'un objet déjà rencontré, auquel cas nous mettons à jour les paramètres de son modèle ; soit nous apprenons un nouvel objet, dans ce cas il faut incrémenter le nombre d'objets connus : $\mathrm{N}_{\mathrm{obj_c}}$.

Par la suite nous nous réfèrerons à cette description par les connaissances préalables notées *CP_DetectObjet* (associée implicitement au jeu de données utilisé pour l'identification) :

$$\mathrm{P}\left(\texttt{Obj_id Dnv Mnv Mld Per} \mid \textit{Cp_DetectObjet}\right) = \begin{vmatrix} \mathrm{P}\left(\texttt{Obj_id} \mid \textit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{Dnv} \mid \texttt{Obj_id}\ \textit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{Mnv} \mid \texttt{Obj_id}\ \textit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{Mld} \mid \texttt{Obj_id}\ \textit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{Per} \mid \texttt{Obj_id}\ \textit{Cp_DetectObjet}\right). \end{vmatrix} \tag{8.18}$$

8.2.4 Utilisation

8.2.4.1 Programme et question probabiliste

Une fois que le Khepera a fait le tour d'un objet à reconnaître, la requête posée afin de déterminer l'objet courant est :

1/ Descriptions utilisées :

Cp_DetectObjet

2/ Requête :

E_q : {Obj_id}

E_c : {Dnv, Mnv, Mld, Per}

E_i : Ø

Question : P(Obj_id | Dnv Mnv Mld Per *Cp_DétecObjet*)

3/ Décision :

Tirage aléatoire

La réponse à cette question est obtenue par :

$$\mathrm{P}\left(\texttt{Obj_id} \mid \texttt{dnv mnv mld per}\ \textit{Cp_DetectObjet}\right) = \frac{\mathrm{P}\left(\texttt{Obj_id dnv mnv mld per} \mid \textit{Cp_DetectObjet}\right)}{\mathrm{P}\left(\texttt{dnv mnv mld per} \mid \textit{Cp_DetectObjet}\right)}. \tag{8.19}$$

Et s'exprime sous la forme d'un produit de quatre distributions pondérées par une constante de normalisation notée Z :

$$\begin{aligned}
&\mathrm{P}\left(\texttt{Obj_id} \mid \texttt{dnv mnv mld per } \mathit{Cp_DetectObjet}\right) \\
&= \frac{\mathrm{P}\left(\texttt{Obj_id dnv mnv mld per} \mid \mathit{Cp_DetectObjet}\right)}{\sum_{\texttt{Obj_id}} \mathrm{P}\left(\texttt{Obj_id dnv mnv mld per} \mid \mathit{Cp_DetectObjet}\right)} \\
&= \frac{1}{Z} \begin{vmatrix} \mathrm{P}\left(\texttt{dnv} \mid \texttt{Obj_id } \mathit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{mnv} \mid \texttt{Obj_id } \mathit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{mld} \mid \texttt{Obj_id } \mathit{Cp_DetectObjet}\right) \\ \mathrm{P}\left(\texttt{Per} \mid \texttt{Obj_id } \mathit{Cp_DetectObjet}\right). \end{vmatrix}
\end{aligned} \tag{8.20}$$

La réponse peut être calculée très efficacement, étant donné qu'il ne s'agit que d'évaluer le produit de quatre distributions très simples.

8.2.4.2 Résultat

Les objets présentés sur la photo (Figure 8-3) sont présentés successivement, 5 fois chacun, dans un ordre aléatoire. À chaque fois on demande au robot : connais-tu cet objet, ou est-il nouveau ?

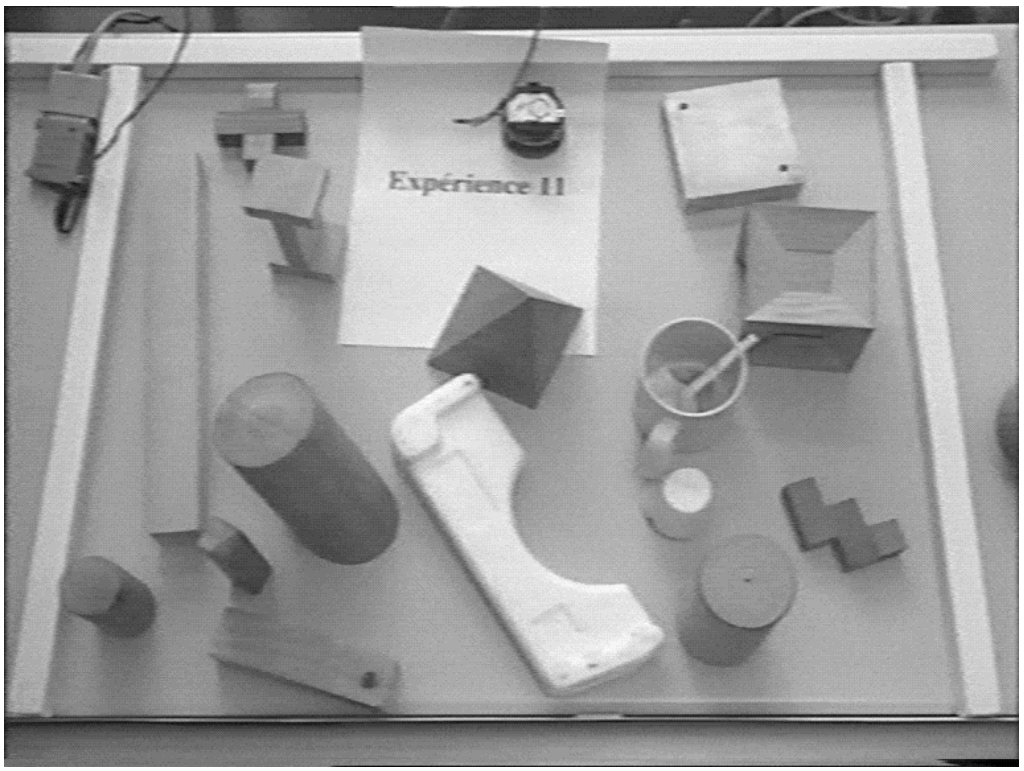


Figure 8-3 — Les différents objets présentés au Khepera.
Si le robot est en présence d'un objet nouveau (i.e. il ne lui a jamais été présenté auparavant) alors il crée un nouvel identificateur d'objet et commence l'apprentissage de cet objet.

Si le robot est en présence d'un objet qu'il pense avoir déjà rencontré, alors, il « reconnaît » l'objet ayant la plus forte probabilité. Il peut, cependant, fournir la probabilité pour tous les objets et donc faire part de ses « doutes ».

Le robot ne s'est jamais trompé pour reconnaître un objet nouveau en tant que tel.

Il n'a quasiment jamais fait d'erreur pour correctement identifier les objets qu'il avait déjà rencontré. Les seules erreurs qui ont été faites ont eu lieu soit au tout début de l'apprentissage d'un objet (1ère présentation) soit sur les deux objets en haut à gauche de l'image qui n'ont jamais pu être distingués l'un de l'autre.

Ce deuxième type d'erreur n'est pas étonnant, les deux objets en question ont strictement la même base carrée et ne diffèrent l'un de l'autre que par leurs hauteurs et leur matière (bois pour l'un et acier pour l'autre). Étant donné les variables utilisées par le Khepera pour reconnaître les objets, il n'a aucune possibilité de distinguer ces deux objets. Il se trouve dans la situation de quelqu'un à qui on demanderait de trouver la couleur d'un objet à tâtons.

8.3 Enseignements et discussion

L'enseignement principal que nous retiendrons de ces expériences est que, finalement, chercher à reconnaître une situation ou un objet peut être vu comme la spécification d'un modèle capteur particulier. Nous avons montré ici que, lors de la spécification de ces descriptions, détecter une situation ou un objet repose sur une démarche identique à celle présentée au Chapitre 5.

Cette démarche peut se résumer de la manière suivante. Soit un objet du monde pour lequel nous choisissons un ensemble de paramètres afin de le décrire et que nous résumons par la variable `V` caractérisant certains invariants physiques de cet objet. Le robot ne peut pas accéder directement à cette variable, c'est-à-dire qu'il ne dispose pas de capteur lui permettant de mesurer directement la valeur de cette variable. Par contre, il dispose de différentes informations sensorielles (`S1`, …, `Sn`). En présence de l'objet, ces informations sensorielles sont influencées par l'objet. Cette influence peut être modélisée dans le sens « causal » habituel, « connaissant le phénomène physique nous pouvons prédire la lecture d'une mesure sensorielle » : P(`Si` | `V` *Ci*). Si nous disposons de chacun de ces modèles élémentaires associé à chaque entrée sensorielle `Si`, en posant l'hypothèse conditionnelle que connaissant complètement le phénomène physique (i.e. la variable `V` est connue), les entrées sensorielles sont supposées indépendantes alors nous pouvons fusionner les différents modèles pour construire une description globale afin d'identifier l'objet :

$$P(\texttt{V S1…Sn} \mid C) = P(\texttt{V} \mid C)P(\texttt{S1} \mid \texttt{V } C1)\ldots P(\texttt{Sn} \mid \texttt{V } Cn)$$

La question posée répond au problème de la perception de bas niveau vu comme un problème d'inversion (voir par exemple dans le cadre de la vision [Poggio84]) :

$$P(\texttt{V} \mid \texttt{S1..Sn } C)$$

L'inférence offre donc un moyen de spécification et de résolution systématique pour le problème de la perception bas niveau.

Chapitre 9

Séquences temporelles

9.1 Objectifs

Dans ce travail, le cadre choisi pour illustrer notre propos est une tâche de surveillance. Le Khepera doit, *in fine*, effectuer un travail de « veilleur de nuit ». Les tâches qu'il a à accomplir sont :

- Patrouiller son environnement et donner l'alarme quand il détecte un mouvement.
- Donner l'alarme et intervenir s'il détecte un « incendie ».
- Reconnaître certains objets si on lui en donne l'ordre.
- Revenir à sa base quand le moment est venu.
- Gérer son énergie en rentrant régulièrement à la base pour se recharger.

Pour réaliser ces tâches, le robot va devoir construire des séquences temporelles de comportements réactifs. L'objectif de ce chapitre est de montrer comment de telles séquences temporelles peuvent être spécifiées dans le cadre probabiliste.

9.2 Séquence temporelle de tâche

9.2.1 Objectifs et protocole expérimental

Au cours de la tâche de surveillance, le Khepera va effectuer successivement cinq types de tâches :

- *Inactif* : Le Khepera est à sa base, il ne bouge pas et attend pour repartir en patrouille qu'on lui en donne l'ordre (`Veille = 1`), et que ces réserves énergétiques (`Eng`) soient suffisantes.
- *Patrouille* : Le Khepera se déplace aléatoirement dans l'environnement en évitant les obstacles. Régulièrement, au cours de la patrouille, il effectue une pose (arrêt) lui permettant de détecter des mouvements suspects à l'aide de sa caméra.
- *Reconnaissance* : Lorsque nous lui en donnons l'ordre, le Khepera effectue le tour d'un objet afin de le reconnaître.
- *Traitement-incendie* : Lorsqu'il détecte un incendie (`Feu = 1`), le Khepera donne l'alarme et tente d'éteindre le feu. Les feux sont matérialisés par des bougies placées dans l'environnement du robot. Pour traiter les incendies, le Khepera dispose d'une micro-turbine qu'il peut mettre en action pour « souffler les bougies ».
- *Retour* : En fin de patrouille (`Veille = 0`) le Khepera retourne à sa base en suivant un comportement de phototaxie tout en évitant les obstacles. La base est une petite niche fortement éclairée par une lampe.
- *Retour-énergie* : Lorsque que ses réserves d'énergie deviennent faibles, le Khepera retourne à sa base afin de recharger ses batteries.

9.2.2 Spécification

9.2.2.1 Choix des variables

Ces différents états sont représentés par la variable `Tach`.

`Tach` : variable à six états {`Inactif`, `Patrouille`, `Reconnaissance`, `Traitement-incendie`, `Retour`, `Retour-énergie`}.

Cette variable est mise en relation avec les variables suivantes :

`Veille` : variable binaire indiquant au Khepera lorsqu'il doit patrouiller ou retourner à sa base et y rester jusqu'à nouvel ordre.

`Eng` : variable à quatre états {`TrèsFort`, `Fort`, `faible`, `très faible`} indiquant le niveau de ses réserves énergétiques.

`Base` : variable binaire valant soit 1 soit 0 selon que le Khepera est à sa base ou non.

`Feu` : variable binaire de détection d'incendie ; l'état de cette variable est déterminé par une analyse de l'image fournie par la caméra linéaire.

`Obj?` : variable binaire permettant de demander au robot de reconnaître l'objet au près duquel il se trouve.

`Tach_t-1` : variable indiquant le type de tâche effectuée au pas de temps précédent.

Nous représentons la dépendance entre ces variables sous la forme d'une description. Par la suite, nous nous y referons par les connaissances préalables qui lui sont associées, dénotées : *Cp_Tach*.

Variables pertinentes

- **Tach** : domaine :{inactif, patrouille, reconnaissance, traitement-incendie, retour, retour-energie}, $k_{\texttt{Tach}} = 6$;
- **Veille** : domaine : {0, 1}, $k_{\texttt{Veille}} = 2$;
- **Eng** : domaine : {TF, F, f, tf}, $k_{\texttt{Eng}} = 4$;
- **Base** : domaine : {0, 1}, $k_{\texttt{Base}} = 2$;
- **Feu** : domaine :{0, 1}, $k_{\texttt{Feu}} = 2$;
- **Obj?** : domaine : {0, 1}, $k_{\texttt{Obj?}} = 2$;
- **Tach_t-1** : domaine :{inactif, patrouille, reconnaissance, traitement-incendie, retour, retour-energie}, $k_{\text{Tach_t-1}} = 6$.

9.2.2.2 Connaissances préalables de dépendance

La distribution conjointe de cette description est :

$$P\left(\texttt{Tach Veille Eng Base Feu Obj? Tach_t-1} \mid Cp_Tach\right). \tag{9.1}$$

À chaque pas de temps, le robot va devoir décider quelle tâche accomplir sachant la tâche que l'on poursuivait à l'instant précédent et sachant les valeurs des variables Veille, Eng, Base, Feu et Obj? qui permettent de choisir entre les différente tâches. La question qui va devoir être posée sera donc :

$$P\left(\texttt{Tach} \mid \texttt{Veille Eng Base Feu Obj? Tach_t-1}\ Cp_Tach\right). \tag{9.2}$$

Il est tentant de chercher à spécifier cette distribution (9.2) directement. Ce serait suivre la démarche de programmation habituelle qui consiste, étant donné les conditions à l'instant *t-1*, à déterminer ce qui doit être fait à l'instant *t*. Cette démarche peut s'avérer complexe car il peut rapidement y avoir un très grand nombre d'états possibles à *t-1* (ici 320=2×4×2×2×2×5, alors que chacune des cinq variables n'a que très peu de cas possibles).

Nous proposons d'adopter une démarche de programmation dite « inverse ». En effet, il est simple, étant donné la tâche, d'avoir une idée sur les valeurs possibles des variables Veille, Eng, Base, Feu et Obj? qui peuvent y correspondre. Par exemple, si le Khepera est en Patrouille c'est qu'il en a reçu l'ordre et donc Veille a nécessairement comme valeur 1. De plus, on peut raisonnablement considérer que ces cinq variables sont indépendantes les unes des autres conditionnellement à la connaissance de la tâche en train de s'accomplir.

Ces considérations nous conduisent à décomposer la distribution conjointe de la manière suivante :

$$\mathrm{P}\left(\texttt{Tach Veille Eng Base Feu Obj? Tach_t-1} \mid Cp_Tach\right) = \begin{vmatrix} \mathrm{P}\left(\texttt{Tach_t-1} \mid Cp_Tach\right) \\ \mathrm{P}\left(\texttt{Tach} \mid \texttt{Tach_t-1}\ Cp_Tach\right) \\ \mathrm{P}\left(\texttt{Veille} \mid \texttt{Tach}\ Cp_Tach\right) \\ \mathrm{P}\left(\texttt{Eng} \mid \texttt{Tach}\ Cp_Tach\right) \\ \mathrm{P}\left(\texttt{Base} \mid \texttt{Tach}\ Cp_Tach\right) \\ \mathrm{P}\left(\texttt{Feu} \mid \texttt{Tach}\ Cp_Tach\right) \\ \mathrm{P}\left(\texttt{Obj?} \mid \texttt{Tach}\ Cp_Tach\right). \end{vmatrix} \tag{9.3}$$

Une telle décomposition offre notamment l'avantage de réduire considérablement la dimension du problème dans la phase de spécification.

9.2.2.3 Connaissances préalables d'observation

Nous choisissons de représenter toutes ces distributions sous formes discrètes. Les différentes valeurs qu'elles peuvent prendre vont être spécifiées une à une.

9.2.3 Identification

Nous spécifions, tout d'abord, l'a priori sur la variable Tach_t-1 sous la forme d'une distribution uniforme :

$$\mathrm{P}\left(\texttt{Tach_t-1} \mid\ Cp_Tach\right) = \mathbf{U}_{k\texttt{Tach_t-1}} = \frac{1}{6}. \tag{9.4}$$

Les valeurs des autres distributions élémentaires sont données sous forme de tables. Ces tables se lisent par colonnes. Par exemple, la première colonne du tableau ci-contre donne la probabilité de Tach sachant que Tach_t-1 est Inactif (P(Tach | [Tach_t-1 = Inactif])). La somme des valeurs d'une même colonne doit donc être égale à 1 afin de respecter la contrainte de normalisation.

Dans ces tableaux, nous utilisons la marque x pour indiquer que nous répartissons la « masse » de probabilité non spécifiée de manière équiprobable sur les différents cas marqués, de telle manière que la somme de tous les cas ait pour valeur 1. Par exemple, dans la première colonne du tableau ci-contre, x vaut 0,05 (0,05 = (1 - 0,9) / 2).

9.2.3.1 P(Tach | Tach_t-1 *Cp_Tach*)

La deuxième distribution que nous spécifions est la probabilité que le Khepera passe dans un certain mode d'action (Tach) sachant qu'au pas précédent il était dans le mode spécifié par la variable Tach_t-1.

P(**Tach** | **Tach_t-1** *Cp_Tach*)						
Tach \ Tach_t-1	inactif	patrou	recon	trait-inc	ret-eng	retour
inactif	0,9	x	0	0	x	x
patrou.	x	0,9	x	x	x	x
reconn.	0	x	0,99	0	x	x
trait-inc	x	x	x	x	x	x
ret-eng	0	x	x	x	0,9	x
retour	0	x	x	0	x	0,9

Nous présentons, ci-dessous, colonne par colonne, les arguments qui nous ont menés à spécifier ces valeurs :

Si au pas de temps précédent, le Khepera est en mode `inactif` (colonne 1) alors nous voulons qu'il ait une bonne chance d'y rester (probabilité 0,9), nous voulons qu'il n'ait aucune possibilité de passer directement de ce mode en mode `reconnaissance`, `retour-énergie` ou `retour` (probabilité 0) et nous lui accordons la possibilité de passer en mode `traitement-incendie` ou `patrouille` (probabilité 0,05 pour chacun des deux cas, obtenue par normalisation)

Si au pas de temps précédent, le Khepera est en mode `patrouille` (colonne 2), nous voulons qu'il reste dans ce mode (probabilité de 0,9), mais nous lui laissons la possibilité de changer pour n'importe quel autre mode.

En mode `reconnaissance` (colonne 3), nous fixons une forte probabilité qu'il reste dans ce mode (0,99) car nous ne voulons pas que le robot puisse facilement se laisser « distraire » de sa tâche de reconnaissance. Nous lui interdisons absolument de passer en mode `inactif` (probabilité 0) et nous lui laissons une faible possibilité de passer dans un des autres modes.

Lorsque le robot est en mode `traitement-incendie` (colonne 4), nous imposons qu'il ne puisse en sortir que pour passer en mode `patrouille` ou en mode `retour-énergie`, la probabilité des autres modes étant fixée à 0.

Lorsque le Khepera est en mode `retour-énergie` (colonne 5), il peut basculer dans n'importe quel autre mode avec une probabilité faible, le plus probable étant qu'il reste dans ce mode (probabilité 0,9). Nous avons spécifié de manière similaire le cas où il est en mode `retour` (colonne 6).

Deux éléments essentiels sont spécifiés dans cette table :

- La « stabilité » (continuer à faire la même chose) du type d'action engagé par le Khepera qui peut être modulée (probabilité de 0,9 ou 0,99).
- L'interdiction de passer de certains modes à certains autres (probabilité 0).

9.2.3.2 P(`Veille` | `Tach` *Cp_Tach*)

Dans le tableau suivant nous donnons la probabilité d'être en `Veille` sachant la tâche en train de s'accomplir :

P(**Veille** | **Tach** *Cp_Tach*)						
Veille \ Tach	inactif	patrou	recon	trait-inc	ret-eng	retour
0	0,9	0	0	x	x	1
1	x	1	1	x	x	0

Lorsque le Khepera est dans le mode `inactif` (colonne 1), le plus probable est qu'il ne soit pas en `veille` (probabilité 0,9). Cependant, il est possible qu'il soit en `veille` mais `inactif` pour cause de chargement de batterie.

Dans les cas où il se trouve soit en `patrouille` (colonne 2), soit en mode `reconnaissance` d'objet (colonne 3), nous considérons qu'il ne peut être qu'en `veille`.

A contrario, s'il est en mode `retour` (colonne 6), c'est nécessairement qu'on lui en a donné l'ordre et donc il ne peut plus être en `veille`.

Les deux autres cas, `traitement-incendie` (colonne 4) et `retour-énergie` (colonne 5), peuvent se produire qu'il soit en `veille` ou non, on ne peut donc rien prédire sur `veille` dans ces cas (probabilité 0,5).

9.2.3.3 P(`Eng` | `Tach` *Cp_Tach*)

Nous spécifions de la même manière les probabilités associées à la variable `Eng`.

P(**Eng** | **Tach** *Cp_Tach*)						
Eng \ Tach	inactif	patrou.	recon.	trait-inc	ret-eng	retour
tf	0,325	0	0	x	0,8	x
f	0,325	0,1	0,1	x	0,2	x
F	0,25	x	x	x	0	x
TF	0,1	x	x	x	0	x

Lorsque le Khepera est `inactif` (colonne 1), nous considérons qu'il est soit en train de recharger ses réserves énergétiques soit que sa mission est suspendue. Dans ce dernier cas, le niveau énergétique peut prendre n'importe quelle valeur. Par contre, lorsqu'il est `inactif` afin de recharger, nous pensons qu'il y a de fortes chances que ses réserves soient `faible` ou `trèsfaible`, un peu moins d'être `Forte` et beaucoup moins d'être `TrèsForte`.

Dans les cas où le robot est mode `patrouille` (colonne 2) ou `reconnaissance` d'objet (colonne 3), nous indiquons qu'il ne peut être dans ces modes lorsque que son énergie est très faible, et qu'il est possible, avec une probabilité relativement faible (0,1), qu'il ait un niveau d'énergie faible.

Lorsqu'il est en mode `retour-energie` (colonne 5), c'est que nécessairement son énergie est `faible` ou `trèsfaible` avec une probabilité supérieure pour ce dernier niveau ; en aucun cas il ne peut avoir un niveau d'énergie `Fort` ou `TrèsFort`.

Pour les deux modes, `traitement-incendie` (colonne 4) et `retour` (colonne 6) nous ne pouvons rien prédire sur cette variable. Nous spécifions ainsi, implicitement, qu'en

cas d'incendie, le niveau énergétique importe peu et que le Khepera doit faire son possible soit pour intervenir soit pour rentrer à la base.

9.2.3.4 P(Base | Tach *Cp_Tach*)

P(Base | Tach *Cp_Tach*)						
Base \ Cpt	inactif	patrou.	recon.	trait-inc	ret-eng	retour
0	0	0,99	0,99	x	0,99	0,99
1	1	0,01	0,01	x	0,01	0,01

Si le Khepera est `inactif` c'est nécessairement qu'il est à sa base.

Dans les modes `patrouille`, `reconnaissance`, `retour-energie` et `retour`, le Khepera ne se trouve, a priori, pas à sa base (`Base` = 0) toutefois cette éventualité n'est pas complètement impossible (probabilité de 0,01). Effectivement, au cours de l'exécution de ces différents modes, il existe une éventualité (même faible) qu'il passe par la base (et plus particulièrement dans les modes `retour` et `retour-energie`).

Lors du traitement d'un incendie nous considérons qu'il peut tout aussi bien être à sa base que ne pas y être.

9.2.3.5 P(Obj? | Tach *Cp_Tach*)

P(Obj? | Tach *Cp_Tach*)						
Obj? \ Tach	inactif	patrou.	recon.	trait-inc	ret-eng	retour
0	x	x	0	x	x	x
1	x	x	1	x	x	x

Ici nous ne spécifions que très peu de chose. Nous disons simplement que lorsque le robot est en mode `reconnaissance` d'objet, alors c'est que nécessairement on lui en a donné l'ordre.

9.2.3.6 P(Feu | Tach *Cp_Tach*)

P(Feu | Tach *Cp_Tach*)						
Feu \ Tach	inactif	patrou.	recon.	trait-inc	ret-eng	retour
0	1	1	1	0	1	1
1	0	0	0	1	0	0

Cette dernière spécification est très simple. Elle nous permet d'indiquer que le Khepera ne peut en aucun cas être dans un mode différent de `traitement-incendie` en présence d'un feu (`Feu` = 1).

9.2.4 Utilisation

9.2.4.1 Programme et question probabiliste

Comme il a été dit au début de la section 9.2.2.2, la requête consiste à trouver la tâche à accomplir connaissant la tâche effectuée au pas de temps précédent et les valeurs des variables sensorielles :

1/ Description utilisée :

Cp_Tach

2/ Requête :

E_q : {Tach}

E_c: {Tach_t−1, Veille, Eng, Base, Feu, Obj?}

E_i: Ø

Question : P(Tach | Tach_t−1 Veille Eng Base Feu Obj? *Cp_Tach*)

3/ Décision :

Tirage aléatoire

La question qui se pose est donc :

$$P(\texttt{Tach} \mid \texttt{Tach_t-1 Veille Eng Base Feu Obj?}\ Cp_Tach). \tag{9.5}$$

La réponse à cette question est donnée par :

$$P(\texttt{Tach} \mid \texttt{Veille Eng Base Feu Obj? Tach_t-1}\, Cp_Tach)$$

$$= \frac{1}{\mathbf{Z}} \begin{vmatrix} P(\texttt{Tach} \mid \texttt{Tach_t-1}\ Cp_Tach) \\ P(\texttt{Veille} \mid \texttt{Tach}\ Cp_Tach) \\ P(\texttt{Eng} \mid \texttt{Tach}\ Cp_Tach) \\ P(\texttt{Base} \mid \texttt{Tach}\ Cp_Tach) \\ P(\texttt{Feu} \mid \texttt{Tach}\ Cp_Tach) \\ P(\texttt{Obj?} \mid \texttt{Tach}\ Cp_Tach), \end{vmatrix} \tag{9.6}$$

où Z est une constante de normalisation qu'il n'est pas nécessaire de calculer explicitement. On voit que le calcul correspondant est très aisé.

9.2.4.2 Résultat

Les résultats correspondants apparaîtront explicitement au chapitre suivant quand nous présenterons l'intégration.

Le Khepera peut ainsi construire une séquence temporelle de tâches. Comme nous le verrons cette séquence temporelle s'avère très satisfaisante pour l'observateur. En particulier, elle est stable, le Khepera n'apparaît pas comme une « girouette » qui change d'avis tout le

temps. La séquence est cependant très souple dans le sens où le Khepera change de tâche sans délais quand il le faut. Enfin, ce mode de construction de séquences temporelles est robuste aux changements de l'environnement dans la mesure où il permet de produire des séquences satisfaisantes pour toute une large gamme d'environnements différents.

9.2.5 Enseignements et discussion

Cette expérience illustre une méthode entièrement nouvelle et originale de construction de séries temporelles de tâches. Cette méthode pourrait être qualifiée de programmation temporelle « inverse ». En effet, on ne spécifie pas, comme il est d'usage, les conditions requises à l'application d'une certaine tâche. On donne, au contraire, pour chaque tâche, les observations associées, en considérant ces observations comme indépendantes les unes des autres sachant la tâche. La combinatoire des cas à envisager s'en trouve considérablement réduite. À l'usage, cette approche, tout d'abord déroutante, s'avère très naturelle et facile à suivre.

9.3 Séquence temporelle de déplacement

9.3.1 Objectifs et protocole expérimental

Nous venons de spécifier sous la forme d'une description, la succession des différentes tâches du Khepera. À chacune de ces tâches va être associé un type de déplacement particulier ou une séquence de divers types de déplacements. Enfin, un type de déplacement va être associé à l'un des comportements réactifs appris précédemment.

Dans ce paragraphe, nous allons spécifier la description permettant d'associer tâches et séquences de types de déplacements.

Nous avons pour cela créé la variable `Td`. Celle-ci représente les différents modes de déplacement dont nous disposons sous forme de comportement réactif : arrêt, évitement d'obstacle, suivi de contour, combinaison phototaxie et évitement, et enfin pompier.

Ce dernier type de déplacement consiste à exécuter un programme classique, décrit autrement que par le biais d'une description probabiliste. Nous entendons montrer ainsi qu'il est possible de mélanger aisément ces deux approches de la programmation des robots.

Rappelons que les tâches que le Khepera doit accomplir sont : `inactif`, `patrouille`, `reconnaissance`, `traitement-incendie`, `retour-énergie` et `retour`.

A la tâche `inactif` est associée le type de déplacement `arrêt`, à la tâche `traitement-incendie` est associée le type de déplacement `pompier`, aux tâches `retour-énergie` et `retour` sont associés le type de déplacement qui est la combinaison de la phototaxie et de l'évitement d'obstacle, `photo-évit`.

Nous souhaitons, qu'en mode `patrouille` le Khepera effectue régulièrement des arrêts afin d'être en mesure de détecter des mouvements suspects dans son champ de vision. Dans ce mode, le Khepera basculera, donc, alternativement du type de déplacement arrêt (`Td = arrêt`) à un mode de déplacement aléatoire évitant le contact avec les obstacles (`Td = évitement`).

Finalement, en mode `reconnaissance`, nous souhaitons que le Khepera effectue un suivi de contour tant que le tour de l'objet n'est pas achevé. Une fois le tour achevé, le robot s'arrête en attendant de passer à une autre tâche. La tâche de reconnaissance correspond donc à la séquence de type de déplacement suivi de contour puis arrêt.

9.3.2 Spécification

9.3.2.1 Choix des variables

Pour réaliser ces deux dernières séquences, nous avons besoin de créer deux nouvelles variables sensorielles, `Tour` et `Tempo`.

La variable `Tour` est très simple, elle indique, une fois la reconnaissance d'objet engagée, si le tour de l'obstacle est effectué. Elle est aisément calculable grâce aux données odométriques du robot.

La variable `Tempo` chronomètre le temps en secondes durant lequel le Khepera est resté dans le même type de déplacement. Nous avons fixé le domaine de cette variable entre 0 et 30 secondes. Pour des durées supérieures, la variable `Tempo` reste à la valeur 30.

Pour être en mesure d'utiliser cette dernière information il est nécessaire de prendre en compte le type de déplacement appliqué au pas de temps précédent mémorisé dans la variable `Td_t-1`.

Variables pertinentes

- **Tach** domaine :{inactif, patrouille, reconnaissance, traitement-incendie, retour, retour-energie}, $k_{\text{Tach}} = 6$;
- **Td** : domaine :{arrêt, évitement, suivi, photo-évit, pompier}, $k_{\text{Td}} = 5$;
- **Tempo** : domaine : {0, 1,…, 30}, $k_{\text{Tempo}} = 31$;
- **Tour** : domaine : {0, 1}, $k_{\text{Tour}} = 2$;
- **Td_t-1** : domaine :{arrêt, évitement, suivi, photo-évit, pompier}, $k_{\text{Td}} = 5$;

9.3.2.2 Connaissances préalables de dépendance

La distribution conjointe de cette description est :

$$P\left(\texttt{Td Tach Tour Tempo Td_t-1} \mid Cp_TypDepl\right). \tag{9.7}$$

Nous décomposons cette expression de la manière suivante :

$$\mathrm{P}\left(\texttt{Td Tach Tour Tempo Td_t-1} \mid Cp_TypDepl\right) = \begin{vmatrix} \mathrm{P}\left(\texttt{Td_t-1} \mid Cp_TypDepl\right) \\ \mathrm{P}\left(\texttt{Tempo} \mid Cp_TypDepl\right) \\ \mathrm{P}\left(\texttt{Tour} \mid Cp_TypDepl\right) \\ \mathrm{P}\left(\texttt{Tach} \mid Cp_TypDepl\right) \\ \mathrm{P}\left(\texttt{Td} \mid \texttt{Tach Tempo Tour Td_t-1}\ Cp_TypDepl\right). \end{vmatrix} \tag{9.8}$$

Par cette décomposition, nous choisissons de considérer les variables `Td_t-1`, `Tempo`, `Tour` et `Tach` indépendantes les unes des autres, la dépendance globale s'exprimant par le lien qu'elles ont avec la variable `Td`.

9.3.2.3 Connaissances préalables d'observation

Nous spécifions les a priori sur les variables `Td_t-1`, `Tempo`, `Tour` et `Cpt` sous la forme de distributions uniformes : dans le cadre de cette description nous ne désirons pas favoriser a priori une valeur particulière de ces variables :

$$\begin{aligned} \mathrm{P}\left(\texttt{Td_t-1} \mid Cp_TypDepl\right) &= \mathrm{U}_{k\texttt{Td_t-1}} = \frac{1}{5}, \\ \mathrm{P}\left(\texttt{Tempo} \mid Cp_TypDepl\right) &= \mathrm{U}_{k\texttt{Tempo}} = \frac{1}{31}, \\ \mathrm{P}\left(\texttt{Tour} \mid Cp_TypDepl\right) &= \mathrm{U}_{k\texttt{Tour}} = \frac{1}{2}, \\ \mathrm{P}\left(\texttt{Tach} \mid Cp_TypDepl\right) &= \mathrm{U}_{k\texttt{Tach}} = \frac{1}{6}. \end{aligned} \tag{9.9}$$

Nous choisissons de représenter la dernière distribution sous forme discrète. Les différentes valeurs vont être spécifiées une à une.

9.3.3 Identification

Il nous reste enfin à spécifier la dernière distribution élémentaire. Nous présentons cette spécification sous la forme de la table suivante :

P(Td | Tach Tempo Tour Td_t-1 *Cp_TypDepl*)

Td\Tach	inactif	patrou.	recon.	trait-inc	ret-eng	retour
arrêt	1	1-P1*	[Tour=1]	0	0	0
évitement	0	P1*	0	0	0	0
suivi	0	0	[Tour=0]	0	0	0
photo-évit	0	0	0	0	1	1
pompier	0	0	0	1	0	0

Les différentes valeurs attribuées au cas où la variable `Tach` indique un mode d'action tel que `inactif`, `traitement-incendie`, `retour-energie` ou `retour`, montrent une forme de spécification que l'on peut qualifier de déterministe. Par exemple, la première colonne indique clairement que si le comportement du robot est `inactif` alors le type de déplacement est nécessairement `arrêt`.

En mode `reconnaissance`, nous indiquons que si le tour de l'objet n'est pas atteint (`Tour` = 0), le Khepera adopte un comportement réactif correspondant au suivi de contour, une fois le tour accompli (`Tour` = 1), le Khepera s'arrête. Plus formellement cela nous donne :

$$\mathrm{P}\left([\texttt{Td} = \texttt{suivi}] \,\middle|\, \begin{matrix}[\texttt{Tach} = \texttt{recon.}]\ \texttt{Tempo} \\ [\texttt{Tour} = 0]\ \texttt{Td_t-1}\end{matrix} \; Cp_TypDepl\right) = 1 \tag{9.10}$$

et :

$$\mathrm{P}\left([\texttt{Td} = \texttt{arrêt}] \,\middle|\, \begin{matrix}[\texttt{Tach} = \texttt{recon.}]\ \texttt{Tempo} \\ [\texttt{Tour} = 1]\ \texttt{Td_t-1}\end{matrix} \; Cp_TypDepl\right) = 1. \tag{9.11}$$

En mode `patrouille` nous définissons la probabilité, notée P1* dans la table, en fonction des valeurs des variables `Tempo` et `Td_t-1` de la manière suivante :

$$\begin{aligned}
&\mathrm{P}\left(\texttt{Td} \mid [\texttt{Tach} = \texttt{patrouille}]\ \texttt{Tour}\ \texttt{Tempo}\ \texttt{Td_t-1}\ Cp_TypDepl\right) \\
&= \mathrm{P}\left(\texttt{Td} \mid [\texttt{Tach} = \texttt{patrouille}]\ \texttt{Tempo}\ \texttt{Td_t-1}\ Cp_TypDepl\right) \\
&= \begin{cases} \texttt{P1*} & \text{pour } [\texttt{Td} = \texttt{évitement}] \\ \texttt{1-P1*} & \text{pour } [\texttt{Td} = \texttt{arrêt}] \\ 0 & \text{pour les autres valeurs de } \texttt{Td} ; \end{cases}
\end{aligned} \tag{9.12}$$

selon `Td_t-1`

$$\texttt{P1*} = \begin{cases} \texttt{P1}_{\texttt{evit}} & \text{si } [\texttt{Td_t-1} = \texttt{évitement}] \\ \texttt{1-P1}_{\texttt{arrêt}} & \text{si } [\texttt{Td_t-1} = \texttt{arrêt}] \\ 1 & \text{sinon.} \end{cases}$$

$\texttt{P1}_{\texttt{evit}}$ représente la probabilité de rester dans le type de déplacement correspondant à l'évitement. Nous souhaitons maintenir ce type déplacement au moins `Temps_mini_évit` et au plus `Temps_maxi_évit`. Entre ces deux valeurs, la probabilité décroît linéairement :

$$P1_{evit} = \begin{cases} 1 \text{ si } Tempo \leq Temps_mini_évit \\ 0 \text{ si } Tempo \geq Temps_maxi_évit \\ 1 - \dfrac{Temps_maxi_évit - (Tempo + Temps_mini_évit)}{(Temps_maxi_évit - Temps_mini_évit)}. \end{cases} \quad (9.13)$$

Réciproquement, $P1_{arrêt}$ représente la probabilité de rester à l'arrêt. Nous souhaitons maintenir le Khepera à l'arrêt au moins `Temps_mini_arrêt` et au plus `Temps_maxi_arrêt`. Entre ces deux valeurs, la probabilité décroît linéairement :

$$P1_{arrêt} = \begin{cases} 1 \text{ si } Tempo \leq Temps_mini_arrêt \\ 0 \text{ si } Tempo \geq Temps_maxi_arrêt \\ 1 - \dfrac{Temps_maxi_arrêt - (Tempo + Temps_mini_arrêt)}{(Temps_maxi_arrêt - Temps_mini_arrêt)}. \end{cases} \quad (9.14)$$

Nous fixons ainsi une durée minimale et maximale spécifique à chacun des types de déplacement associés au mode patrouille, avec une décroissance linéaire entre les deux bornes. Dans les cas où le type de déplacement précédent n'est ni `évitement` ni `arrêt`, nous imposons d'initialiser la séquence en commençant par `évitement`.

9.3.4 Utilisation

9.3.4.1 Programme et question probabiliste

L'utilisation de cette description nous permet de déterminer le type de déplacement que doit suivre le Khepera. La requête correspondante est la suivante :

1/ Description utilisée :

Cp_TypDepl

2/ Requête :

E_q : {Td}

E_c: {Tach, Tempo, Tour Td_t-1}

E_i: Ø

Question : P(Td | Tach Tempo Tour Td_t-1 *Cp_TypDepl*)

3/ Décision :

Tirage aléatoire

La réponse est obtenue directement à partir de la spécification des connaissances préalables, la valeur obtenue pour la variable `Td` sera mémorisée dans la variable `Td_t-1` pour la question au pas de temps suivant.

9.3.4.2 Résultat

Les résultats seront explicités dans le chapitre suivant.

9.3.5 Enseignements et discussion

Cette expérience illustre comment nous pouvons obtenir de manière probabiliste des séquences temporelles plus « classiques » que dans le paragraphe précédent (section 9.2).

En effet, le fonctionnement obtenu avec cette description est très proche de celui d'un automate. La seule différence notable étant que le choix entre les types de déplacements `arrêt` et `évitement` lors de la tâche de patrouille se fait de manière non déterministe, dans la plage temporelle entre `Temps_mini_évit` et `Temps_maxi_évit` pour le passage de l'évitement à l'arrêt et, réciproquement, entre `Temps_mini_arrêt` et `Temps_maxi_arrêt` pour le passage de l'arrêt à l'évitement.

Chapitre 10

Intégration : une tâche de surveillance

10.1 Objectifs

Dans cette dernière expérience, notre objectif est tout d'abord de montrer comment l'expression d'une tâche complexe robotique peut être obtenue sous la forme d'une description. Nous montrons également que cette description peut être construite sous la forme d'une « structuration » de descriptions plus élémentaires, et en particulier nous illustrons comment nous pouvons utiliser les différentes descriptions, spécifiées tout au long de ces chapitres, comme briques de base pour programmer.

Rappelons que le Khepera doit accomplir les tâches suivantes :

- Patrouiller son environnement et donner l'alarme quand il détecte un mouvement.
- Donner l'alarme et intervenir s'il détecte un « incendie ».
- Reconnaître certains objets si on lui en donne l'ordre.
- Revenir à sa base quand le moment est venu.
- Gérer son énergie en rentrant régulièrement à la base pour se recharger.

10.2 Les descriptions disponibles

Rappelons, tout d'abord, les différentes descriptions à notre disposition.

10.2.1 Capteurs

Nous avons une description permettant de déterminer la position d'une source lumineuse (Chapitre 5) :

$$\mathrm{P}(\texttt{ThetaL DistL Lm0..Lm7} \mid \mathit{Cp_SL}). \tag{10.1}$$

Nous avons ensuite une description pour savoir si on est à la base (section 8.1)

$$\mathrm{P}(\texttt{Base Lm0K Lm7 Px0K Px7} \mid \mathit{Cp_DetectBase}). \tag{10.2}$$

Au paragraphe 8.2 a été définie une description pour faire la reconnaissance des objets :

$$\mathrm{P}(\texttt{Obj_id Dnv Mnv Mld Per} \mid \mathit{Cp_DetectObj}). \tag{10.3}$$

10.2.2 Comportements réactifs

Aux chapitres 4, 6 et 7, ont été définies des descriptions permettant à Khepera d'exécuter un certain nombre de comportements réactifs :

- Évitement d'obstacle (section 4.3 et 6.1) :

$$\mathrm{P}(\texttt{Vrot Vtrans Vtrans_c Dir Prox} \mid \mathit{Cp_Evit}). \tag{10.4}$$

- Suivi de contour (section 4.2) :

$$\mathrm{P}(\texttt{Vrot Vtrans DirG ProxG} \mid \mathit{Cp_Suivi}). \tag{10.5}$$

- Composition entre phototaxie et évitement d'obstacles (section 6.2) :

$$\mathrm{P}(\texttt{Vrot Vtrans H Dir Prox ThetaL} \mid \mathit{Cp_PhotoEvit}). \tag{10.6}$$

À cela, viennent s'ajouter deux descriptions correspondant à des « programmes classiques ».

- Une description pour l'arrêt :

$$\mathrm{P}(\texttt{Vrot Vtrans} \mid \mathit{Cp_Arrêt}). \tag{10.7}$$

qui impose que les vitesses `Vrot` et `Vtrans` soient nulles.

- Une description pour le comportement pompier :

$$\mathrm{P}(\texttt{Vrot Vtrans} \mid \mathit{Cp_Pompier}). \tag{10.8}$$

qui impose que les vitesses `Vrot` et `Vtrans` prennent les valeurs que détermine le programme pompier.

10.2.3 Déplacements

Les déplacements sont déterminés par la description du paragraphe 9.3 :

$$\mathrm{P}(\texttt{Td Tach Tour Tempo Td_t-1} \mid \mathit{Cp_TypDepl}). \tag{10.9}$$

10.2.4 Tâches

Enfin, la séquence temporelle de tâches sera obtenue par la description du paragraphe 9.2 :

$$P(\texttt{Tach Tach_t-1 Veille Eng Base Feu Obj?} \mid Cp_Tach). \quad (10.10)$$

10.3 Spécification

La description globale va représenter notre spécification probabiliste de la tâche de surveillance. Nous utilisons la notation *Cp_Surveil* pour dénoter les connaissances préalables associées à cette description. Par ces connaissances, nous allons indiquer de quelle manière nous combinons les différentes descriptions définies plus haut. Les différents choix que représentent ces connaissances préalables doivent être appréhendés plus comme l'expression de choix permettant d'obtenir un comportement global correspondant aux objectifs que nous nous sommes fixés plutôt qu'une expression d'hypothèses décrivant un quelconque phénomène. C'est en quelque sorte une démarche résolument subjective, ces différents a priori n'ont de sens que relativement aux objectifs propres de cette tâche, et, en aucune sorte, ne prétendent décrire une « vérité-réalité ».

10.3.1 Choix des variables

Les variables retenues se résument à l'ensemble des 42 variables utilisées dans les différentes descriptions composant la description globale. Nous les présentons toutefois sous la forme d'une partition en trois sous-ensembles, $\mathbf{E_s}$ les variables sensorielles (celles dont les valeurs seront connues à tout instant), $\mathbf{E_i}$ les variables intermédiaires (celles dont nous ne connaîtrons pas la valeur, ce sont des variables internes) et enfin $\mathbf{E_m}$ les variables motrices (les variables pour lesquelles nous devrons fixer les valeurs) :

$\mathbf{E_s}$ = {Px0,...Px7, Lm0,...Lm7, Veille, Feu, Obj?, Veille, Eng, Tach_t-1, Td_t-1, Tempo, Tour, Prox, Dir, ProxG, DirG, Dnv, Mnv, Mld, Per} ;

$\mathbf{E_i}$ = {Base, ThetaL, DistL, Tach, Td, H} ;

$\mathbf{E_m}$ = {Vrot, Vtrans, Obj_id}.

10.3.2 Connaissances préalables de dépendances

La distribution conjointe de cette description est :

$$\mathrm{P}\left(\begin{array}{l}\texttt{Px0K Px7 Lm0K Lm7 Veille Feu Obj?}\\ \texttt{Eng Tach_t-1 Td_t-1 Tempo Tour}\\ \texttt{Dir Prox DirG ProxG Vtrans_c}\\ \texttt{Dnv Mnv Mld Per}\\ \texttt{Base ThetaL DistL Tach Td H}\\ \texttt{Vrot Vtrans Obj_id}\end{array}\middle|\; Cp_Surveil\right). \tag{10.11}$$

Nous décomposons l'expression de la distribution conjointe de la manière suivante :

$$\begin{aligned}&\mathrm{P}\left(\begin{array}{l}\texttt{Px0..Px7 Lm0..Lm7 Veille Feu Obj?}\\ \texttt{Eng Tach_t-1 Td_t-1 Tempo Tour}\\ \texttt{Dir Prox DirG ProxG Vtrans_c}\\ \texttt{Dnv Mnv Mld Per}\\ \texttt{Base ThetaL DistL Tach Td H}\\ \texttt{Vrot Vtrans Obj_id}\end{array}\middle|\; Cp_Surveil\right)\\ &\left|\begin{array}{l}\mathrm{P}\left(\begin{array}{l}\texttt{Px0..Px7 Lm0..Lm7 Veille Feu Obj?}\\ \texttt{Eng Tach_t-1 Td_t-1 Tempo Tour}\\ \texttt{Dir Prox DirG ProxG Vtrans_c}\\ \texttt{Dnv Mnv Mld Per}\end{array}\middle|\; Cp_Surveil\right)\\ \mathrm{P}(\texttt{Base} \mid \texttt{Lm0..Lm7 Px0..Px7}\ Cp_Surveil)\\ \mathrm{P}(\texttt{ThetaL DistL} \mid \texttt{Lm0..Lm7}\ Cp_Surveil)\\ \mathrm{P}(\texttt{Tach} \mid \texttt{Veille Eng Base Feu Obj? Tach_t-1}\ Cp_Surveil)\\ \mathrm{P}(\texttt{Td} \mid \texttt{Tach Tempo Tour Td_t-1}\ Cp_Surveil)\\ \mathrm{P}(\texttt{H} \mid \texttt{Prox}\ Cp_Surveil)\\ \mathrm{P}\left(\begin{array}{l}\texttt{Vtrot Vtrans}\end{array}\middle|\begin{array}{l}\texttt{Td H Dir Prox DirG ProxG}\\ \texttt{Vtrans_c ThetaL}\end{array}\ Cp_Surveil\right)\\ \mathrm{P}(\texttt{Obj_id} \mid \texttt{Dnv Mnv Mld Per}\ Cp_Surveil).\end{array}\right.\end{aligned} \tag{10.12}$$

10.3.3 Connaissances préalables d'observation

Les connaissances préalables d'observation se résument à fixer un a priori uniforme pour la distribution des variables « sensorielles » appartenant à $\mathbf{E}_s$. Les autres distributions élémentaires seront calculées par le biais de questions probabilistes posées à des descriptions définies préalablement (voir section suivant 10.4).

10.4 Identification

10.4.1 P(Vs | *Cp_Surveil*)

Nous ne voulons pas exprimer de connaissance particulière a priori sur les variables sensorielles, la distribution sur ces variables est donc fixée uniforme :

$$\mathrm{P}\left(\begin{array}{l}\texttt{Px0..Px7 Lm0..Lm7 Veille Feu Obj?}\\ \texttt{Eng Tach_t-1 Td_t-1 Tempo Tour}\\ \texttt{Dir Prox DirG ProxG Vtrans_c}\\ \texttt{Dnv Mnv Mld Per}\end{array}\middle|\ \mathit{Cp_Surveil}\right)=\mathbf{U}. \tag{10.13}$$

10.4.2 P(Base | Lm0…Lm7 Px0…Px7 *Cp_Surveil*)

La valeur de la variable Base ne dépend que des valeurs des capteurs de proximité et des capteurs de luminosité. Elle est obtenue par le biais d'une question posée à la description *Cp_DetectBase* (section 8.1) :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Base}\mid\texttt{Lm0..Lm7 Px0..Px7}\ \mathit{Cp_Surveil}\right)\\&=\mathrm{P}\left(\texttt{Base}\mid\texttt{Lm0..Lm7 Px0..Px7}\ \mathit{Cp_DétectBase}\right).\end{aligned} \tag{10.14}$$

10.4.3 P(ThetaL DistL | Lm0…Lm7 *Cp_Surveil*)

Les valeurs des ThetaL et DistL ne dépendent que des valeurs des capteurs de luminosité. Elle sont obtenues par le biais d'une question posée à la description *Cp_SL* (section 5.3) :

$$\begin{aligned}&\mathrm{P}\left(\texttt{ThetaL DistL}\mid\texttt{Lm0..Lm7}\ \mathit{Cp_Surveil}\right)\\&=\mathrm{P}\left(\texttt{ThetaL DistL}\mid\texttt{Lm0..Lm7}\ \mathit{Cp_SL}\right).\end{aligned} \tag{10.15}$$

10.4.4 P(Tach | Veille Eng Base Feu Obj? Tach_t-1 *Cp_Surveil*)

La valeur de la variable Tach est obtenue également par le biais d'une question posée à la description *Cp_Tach* (section 9.2) :

$$\begin{aligned}&\mathrm{P}\left(\texttt{Tach}\mid\texttt{Veille Eng Base Feu Obj? Tach_t-1}\ \mathit{Cp_Surveil}\right)\\&=\mathrm{P}\left(\texttt{Tach}\mid\texttt{Veille Eng Base Feu Obj? Tach_t-1}\ \mathit{Cp_Tach}\right).\end{aligned} \tag{10.16}$$

10.4.5 P(Td | Tach Tempo Tour Td_t-1 *Cp_Surveil*)

La valeur de la variable Td est obtenue grâce à la description *Cp_TypDepl* (section 9.3) :

$$\begin{aligned} &\mathrm{P}(\texttt{Td} \mid \texttt{Tach Tempo Tour Td_t-1}\ \mathit{Cp_Surveil}) \\ &= \mathrm{P}(\texttt{Td} \mid \texttt{Tach Tempo Tour Td_t-1}\ \mathit{Cp_TypDepl}). \end{aligned} \tag{10.17}$$

10.4.6 P(H | Prox *Cp_Surveil*)

La valeur de la variable H est obtenue grâce à la description *Cp_PhotoEvit* (section 6.2) :

$$\mathrm{P}(\texttt{H} \mid \texttt{Prox}\ \mathit{Cp_Surveil}) = \mathrm{P}(\texttt{H} \mid \texttt{Prox}\ \mathit{Cp_PhotoEvit}). \tag{10.18}$$

10.4.7 P(Vrot Vtrans | Td H ... *Cp_Surveil*)

Cette distribution qui n'a pas été spécifiée jusqu'à présent permet de faire le lien entre les différents types de déplacements et les comportements réactifs, nous la spécifions selon les valeurs de la variable Td :

$$\mathrm{P}\left(\texttt{Vtrot Vtrans} \,\middle|\, \begin{matrix}[\texttt{Td = arrêt}]\ \texttt{H Dir Prox} \\ \texttt{DirG ProxG Vtrans_c ThetaL}\end{matrix}\ \mathit{Cp_Surveil}\right) \tag{10.19}$$
$$= \mathrm{P}(\texttt{Vtrot Vtrans} \mid \mathit{Cp_Arrêt})\ ;$$

$$\mathrm{P}\left(\texttt{Vtrot Vtrans} \,\middle|\, \begin{matrix}[\texttt{Td = évitement}]\ \texttt{H Dir Prox} \\ \texttt{DirG ProxG Vtrans_c ThetaL}\end{matrix}\ \mathit{Cp_Surveil}\right) \tag{10.20}$$
$$= \mathrm{P}(\texttt{Vtrot Vtrans} \mid \texttt{Dir Prox Vtrans_c}\ \mathit{Cp_Evit})\ ;$$

$$\mathrm{P}\left(\texttt{Vtrot Vtrans} \,\middle|\, \begin{matrix}[\texttt{Td = suivi}]\ \texttt{H Dir Prox} \\ \texttt{DirG ProxG Vtrans_c ThetaL}\end{matrix}\ \mathit{Cp_Surveil}\right) \tag{10.21}$$
$$= \mathrm{P}(\texttt{Vtrot Vtrans} \mid \texttt{DirG ProxG}\ \mathit{Cp_Suivi})\ ;$$

$$\mathrm{P}\left(\texttt{Vtrot Vtrans} \,\middle|\, \begin{matrix}[\texttt{Td = photo-evit}]\ \texttt{H Dir Prox} \\ \texttt{DirG ProxG Vtrans_c ThetaL}\end{matrix}\ \mathit{Cp_Surveil}\right) \tag{10.22}$$
$$= \mathrm{P}(\texttt{Vtrot Vtrans} \mid \texttt{H Dir Prox Vtrans_c ThetaL}\ \ \mathit{Cp_PhotoEvit})\ ;$$

$$\mathrm{P}\left(\texttt{Vtrot Vtrans} \,\middle|\, \begin{matrix}[\texttt{Td = pompier}]\ \texttt{H Dir Prox} \\ \texttt{DirG ProxG Vtrans_c ThetaL}\end{matrix}\ \mathit{Cp_Surveil}\right) \tag{10.23}$$
$$= \mathrm{P}(\texttt{Vtrot Vtrans} \mid \mathit{Cp_Pompier}).$$

10.4.8 P(Obj_id | Dnv Mnv Mld Per *Cp_Surveil*)

La valeur de Obj_id est obtenue grâce à la description *Cp_DetectObj* (section 8.2) :

$$\begin{aligned} &P\left(\texttt{Obj_id} \mid \texttt{Dnv Mnv Mld Per } \textit{Cp_Surveil}\right) \\ &= P\left(\texttt{Obj_id} \mid \texttt{Dnv Mnv Mld Per } \textit{Cp_DétectObj}\right). \end{aligned} \tag{10.24}$$

10.5 Utilisation

10.5.1 Programme et question probabiliste

L'utilisation de cette description va se résumer à déterminer les valeurs de variables motrices `Vrot` et `Vtrans` connaissant les valeurs des variables sensorielles :

1/ Descriptions utilisées :

Cp_Surveil, Cp_TypDepl, Cp_Tach, Cp_Pompier, Cp_Arrêt, Cp_Suivi, Cp_PhotoEvit, Cp_PhotoSL, Cp_Evit, Cp_SL, Cp_DetectBase, Cp_DetectObj.

2/ Requête :

E_q : {Vrot, Vtrans, Obj_id}

E_c: {Px0…Px7, Lm0…Lm7, Veille, Feu, Obj?, Eng, Tach_t-1, Td_t-1, Tempo, Tour, Dir, Prox, DirG, ProxG, Vtrans_c, Dnv, Mnv, Mld, Per}

E_i: {Base, ThetaL, DistL, Tach, Td, H}

Question : P(Vrot, Vtrans, Obj_id | Px0…Px7 … Mld Per *Cp_Surveil*)

3/ Décision :

Tirage aléatoire

Pour cela nous devons calculer la distribution de probabilité sur les variables motrices connaissant les variables sensorielles. Ce calcul s'obtient sous la forme de la question probabiliste suivante :

$$P\left(\texttt{Vrot Vtrans Obj_id} \;\middle|\; \begin{array}{l}\texttt{px0..px7 lm0..lm7 veille feu obj?}\\ \texttt{eng tach_t-1 td_t-1 tempo tour}\\ \texttt{dir prox dirG proxG vtrans_c}\\ \texttt{dnv mnv mld per}\end{array} \textit{Cp_Surveil}\right). \tag{10.25}$$

Nous notons les variables sensorielles en minuscule pour indiquer que, lors de la question, elles ont une valeur particulière qui est connue.

Cette requête est résolue systématiquement par marginalisation sur l'ensemble des variables ignorées $\mathbf{E_i}$= {Base, ThetaL, DistL, Tach, Td, H} :

$$\mathrm{P}\left(\texttt{Vrot Vtrans Obj_id} \;\middle|\; \begin{array}{l}\texttt{px0..px7 lm0..lm7 veille feu obj?}\\ \texttt{eng tach_t-1 td_t-1 tempo tour}\\ \texttt{dir prox dirG proxG vtrans_c}\\ \texttt{dnv mnv mld per}\end{array} \mathit{Cp_Surveil}\right)$$

$$= \frac{1}{Z}\sum_{\texttt{Vi}\in\texttt{Ei}} \mathrm{P}\left(\begin{array}{l}\texttt{Vrot Vtrans Obj_id}\\ \texttt{px0..px7 lm0..lm7 veille feu obj?}\\ \texttt{eng tach_t-1 td_t-1 tempo tour}\\ \texttt{dir prox dirG proxG vtrans_c}\\ \texttt{dnv mnv mld per}\\ \texttt{H Cpt Td ThetaL DistL Base}\end{array} \;\middle|\; \mathit{Cp_Surveil}\right). \tag{10.26}$$

En remplaçant la distribution conjointe par sa décomposition et en effectuant les simplifications évidentes, on obtient :

$$\mathrm{P}\left(\texttt{Vrot Vtrans Obj_id} \;\middle|\; \begin{array}{l}\texttt{px0..px7 lm0..lm7 veille feu obj?}\\ \texttt{eng tach_t-1 td_t-1 tempo tour}\\ \texttt{dir prox dirG proxG vtrans_c}\\ \texttt{dnv mnv mld per}\end{array} \mathit{Cp_Surveil}\right)$$

$$= \frac{1}{Z}\left|\begin{array}{l}\mathrm{P}\left(\texttt{Obj_id} \mid \texttt{dnv mnv mld per}\ \mathit{Cp_DétectObj}\right)\\ \displaystyle\sum_{\substack{\texttt{Td}\\ \texttt{ThetaL}\\ \texttt{H}}}\left(\begin{array}{l}\displaystyle\sum_{\texttt{Tach}}\left(\mathrm{P}\left(\texttt{Td} \;\middle|\; \begin{array}{l}\texttt{Tach}\\ \texttt{td_t - 1 tempo tour}\end{array} \mathit{Cp_TypDépl}\right)\right.\\ \qquad\left.\displaystyle\sum_{\texttt{Base}}\left(\mathrm{P}\left(\texttt{Tach} \;\middle|\; \begin{array}{l}\texttt{Base}\\ \texttt{veille feu obj?}\\ \texttt{eng tach_t - 1}\end{array} \mathit{Cp_Tach}\right)\right.\right.\\ \qquad\left.\left.\mathrm{P}\left(\texttt{Base} \;\middle|\; \begin{array}{l}\texttt{px0...px7}\\ \texttt{lm0...lm7}\end{array} \mathit{Cp_DétectBase}\right)\right)\right)\\ \displaystyle\sum_{\texttt{DistL}}\mathrm{P}\left(\texttt{ThetaL DistL} \mid \texttt{lm0..lm7}\ \ \mathit{Cp_SL}\right)\\ \mathrm{P}\left(\texttt{H} \mid \texttt{prox}\ \mathit{Cp_Surveil}\right)\\ \mathrm{P}\left(\texttt{Vrot Vtrans} \;\middle|\; \begin{array}{l}\texttt{H Td ThetaL}\\ \texttt{dir prox dirG proxG vtrans_c}\end{array} \mathit{Cp_Surveil}\right)\end{array}\right|. \tag{10.27}$$

Cette expression peut paraître relativement complexe. En fait, elle reflète exactement la structure du raisonnement à suivre pour résoudre le problème posé :

- La question sur la reconnaissance des objets est totalement indépendante du problème du contrôle du Khepera.
- La somme la plus interne :

$$\sum_{\texttt{Base}}\left(\begin{array}{l}\mathrm{P}\left(\texttt{Tach} \;\middle|\; \begin{array}{l}\texttt{Base}\\ \texttt{veille feu obj?}\\ \texttt{eng tach_t - 1}\end{array} \mathit{Cp_Tach}\right)\\ \mathrm{P}\left(\texttt{Base} \;\middle|\; \begin{array}{l}\texttt{px0..px7}\\ \texttt{lm0..lm7}\end{array} \mathit{Cp_DétectBase}\right)\end{array}\right), \tag{10.28}$$

consiste à déterminer la tâche ne connaissant pas la valeur de la variable `Base`.

Ce qui revient à se poser la question suivante :

$$\mathrm{P}\left(\mathtt{Tach} \;\middle|\; \begin{array}{l}\texttt{veille feu obj? eng tach_t - 1}\\ \texttt{px0..px7 lm0..lm7}\end{array} \; Cp_Surveil\right). \tag{10.29}$$

- La somme immédiatement au-dessus :

$$\sum_{\mathtt{Tach}} \left(\begin{array}{l} \mathrm{P}\left(\mathtt{Td} \;\middle|\; \begin{array}{l}\mathtt{Tach}\\ \texttt{td_t - 1 tempo tour}\end{array} \; Cp_TypDépl\right) \\ \sum_{\mathtt{Base}} \left(\begin{array}{l} \mathrm{P}\left(\mathtt{Tach} \;\middle|\; \begin{array}{l}\mathtt{Base}\\ \texttt{veille feu obj?}\\ \texttt{eng tach_t - 1}\end{array} \; Cp_Tach\right) \\ \mathrm{P}\left(\mathtt{Base} \;\middle|\; \begin{array}{l}\texttt{px0..px7}\\ \texttt{lm0..lm7}\end{array} \; Cp_DétectBase\right) \end{array} \right) \end{array} \right), \tag{10.30}$$

qui est équivalente d'après (10.29) à :

$$\sum_{\mathtt{Tach}} \left(\begin{array}{l} \mathrm{P}\left(\mathtt{Td} \;\middle|\; \begin{array}{l}\mathtt{Tach}\\ \texttt{td_t - 1 tempo tour}\end{array} \; Cp_TypDépl\right) \\ \mathrm{P}\left(\mathtt{Tach} \;\middle|\; \begin{array}{l}\texttt{veille feu obj? eng tach_t - 1}\\ \texttt{px0..px7 lm0..lm7}\end{array} \; Cp_Surveil\right) \end{array} \right), \tag{10.31}$$

consiste à déterminer le type de déplacement ne connaissant pour la variable `Tach` que la distribution (10.29). Cela revient à poser la question suivante :

$$\mathrm{P}\left(\mathtt{Td} \;\middle|\; \begin{array}{l}\texttt{td_t - 1 tempo tour}\\ \texttt{veille feu obj? eng tach_t - 1}\\ \texttt{px0..px7 lm0..lm7}\end{array} \; Cp_Surveil\right). \tag{10.32}$$

- De même, la somme sur DistL :

$$\sum_{\mathtt{DistL}} \mathrm{P}\left(\mathtt{ThetaL}\ \mathtt{DistL} \mid \texttt{lm0..lm7} \;\; Cp_SL\right) \tag{10.33}$$

consiste à déterminer la valeur de `ThetaL` ne connaissant pas la valeur de `DistL` et revient à poser la question suivante :

$$\mathrm{P}\left(\mathtt{ThetaL} \mid \texttt{lm0..lm7} \;\; Cp_Surveil\right). \tag{10.34}$$

- Finalement, la somme la plus externe, équivalente d'après (10.32) et (10.34) à :

$$\sum_{\substack{\mathbf{Td}\\ \mathbf{ThetaL}\\ \mathbf{H}}} \left(\begin{array}{l} \mathrm{P}\left(\mathbf{Td} \;\middle|\; \begin{array}{l} \texttt{td_t-1 tempo tour} \\ \texttt{veille feu obj? eng tach_t-1} \quad Cp_Surveil \\ \texttt{px0..px7 lm0..lm7} \end{array} \right) \\ \mathrm{P}\left(\mathbf{ThetaL} \mid \texttt{lm0..lm7} \quad Cp_Surveil \right) \\ \mathrm{P}\left(\mathbf{H} \mid \texttt{prox}\, Cp_Surveil \right) \\ \mathrm{P}\left(\texttt{Vrot Vtrans} \;\middle|\; \begin{array}{l} \mathbf{H\ Td\ ThetaL} \\ \texttt{dir prox dirG proxG vtrans_c} \end{array} Cp_Surveil \right) \end{array} \right). \tag{10.35}$$

consiste à déterminer `Vrot` et `Vtrans` sans avoir fait de choix définitif sur aucune des variables intermédiaires, appartenant à $\mathbf{E}_i$= {`Base`, `ThetaL`, `DistL`, `Tach`, `Td`, `H`}, et en particulier sans avoir décidé quelle tâche ou quel type de déplacement adopter, de quelle direction provient la source lumineuse et quelle est la valeur de la variable `H`.

Ainsi, la question cherchant la vitesse de rotation et de translation résulte de l'imbrication hiérarchique de différentes questions que nous résumons ci-dessous :

$$\mathrm{P}(\mathbf{Base} \mid C1) = \mathrm{P}\left(\mathbf{Base} \;\middle|\; \begin{array}{l} \texttt{px0..px7} \\ \texttt{lm0..lm7} \end{array} \quad Cp_DétectBase \right)$$

$$\mathrm{P}(\mathbf{Tach} \mid C2) = \sum_{\mathbf{Base}} \left(\begin{array}{l} \mathrm{P}\left(\mathbf{Tach} \;\middle|\; \begin{array}{l} \mathbf{Base} \\ \texttt{veille feu obj?} \quad Cp_Tach \\ \texttt{eng tach_t-1} \end{array} \right) \\ \mathrm{P}(\mathbf{Base} \mid C1) \end{array} \right)$$

$$\mathrm{P}(\mathbf{Td} \mid C3) = \sum_{\mathbf{Tach}} \left(\begin{array}{l} \mathrm{P}\left(\mathbf{Td} \;\middle|\; \begin{array}{l} \mathbf{Tach} \\ \texttt{td_t-1 tempo tour} \end{array} Cp_TypDépl \right) \\ \mathrm{P}(\mathbf{Tach} \mid C2) \end{array} \right)$$

$$\begin{aligned} \mathrm{P}(\mathbf{ThetaL} \mid C4) &= \sum_{\mathbf{DistL}} \mathrm{P}(\mathbf{ThetaL\ DistL} \mid \texttt{lm0..lm7} \quad Cp_SL) \\ &= \mathrm{P}(\mathbf{ThetaL} \mid \texttt{lm0..lm7} \quad Cp_SL) \end{aligned} \tag{10.36}$$

$$\mathrm{P}(\mathbf{H} \mid C5) = \mathrm{P}(\mathbf{H} \mid \texttt{prox}\, Cp_PhotoEvit)$$

$$\mathrm{P}(\texttt{Vrot Vtrans} \mid C6) = \sum_{\substack{\mathbf{H}\\ \mathbf{ThetaL}\\ \mathbf{Td}}} \left(\begin{array}{l} \mathrm{P}(\mathbf{Td} \mid C3) \\ \mathrm{P}(\mathbf{ThetaL} \mid C4) \\ \mathrm{P}(\mathbf{H} \mid C5) \\ \mathrm{P}\left(\texttt{Vrot Vtrans} \;\middle|\; \begin{array}{l} \mathbf{H\ Td\ ThetaL} \\ \texttt{dir prox dirG} \quad Cp_Surveil \\ \texttt{proxG vtrans_c} \end{array} \right) \end{array} \right).$$

Pour récapituler, le fil du raisonnement comme l'illustre la Figure 10-1 est donc :

- Chercher si on se trouve ou non à la base : P(`Base` | *C1*).
- Chercher la tâche : P(`Tach` | *C2*).
- Chercher le type de déplacement, la direction de la source lumineuse et la valeur de la variable de composition `H` : P(`Td` | *C3*), P(`TethaL` | *C4*), P(`H` | *C5*).
- Finalement, chercher `Vrot` et `Vtrans` pour contrôler le Khepera : P(`Vrot Vtrans` | *C6*).

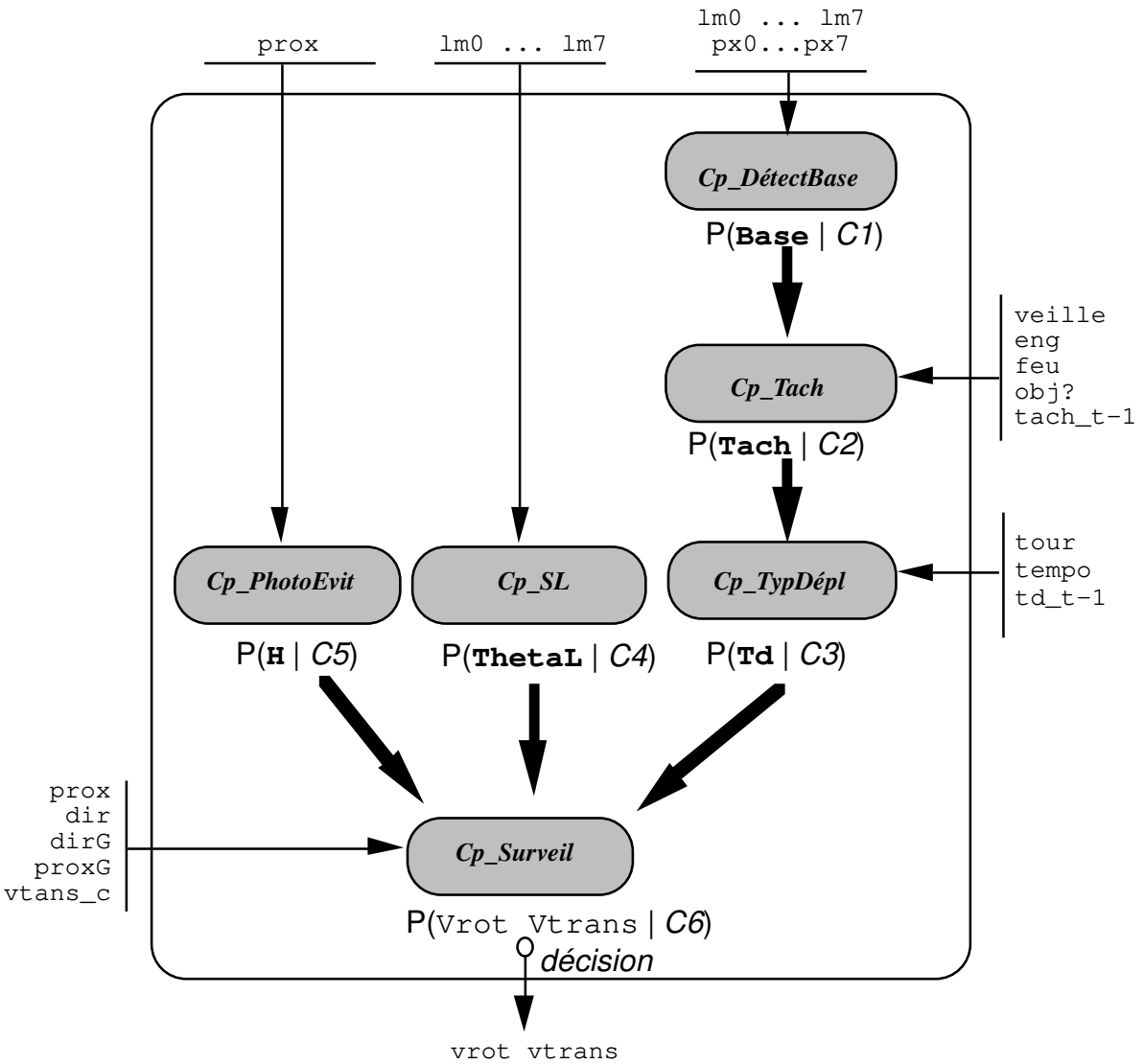


Figure 10-1 — Structure de la question permettant d'inférer `Vrot` et `Vtrans`.
Cette figure illustre l'enchaînement hiérarchique des différentes questions intermédiaires, les flèches noires en gras indiquent les propagations de l'incertitude sur les variables intermédiaires.

Aucune décision n'étant faite sur les variables intermédiaires, l'incertitude est propagée. Le robot adopte un comportement moteur qui est effectivement le mélange probabiliste de plusieurs tâches et de plusieurs types de déplacements.

Cette prise en compte de toute l'information disponible se paye par le coût de calcul de l'évaluation de toutes les sommes.

Cependant, pour n'importe laquelle de ces sommes, nous avons la possibilité de faire le choix de simplifier le calcul, de laisser de côté une partie de l'information, en faisant une décision.

Ainsi, par exemple, nous pouvons remplacer la somme :

$$\sum_{\mathtt{Base}} \left(\mathrm{P}\left(\mathtt{Tach} \;\middle|\; \begin{array}{l} \mathtt{Base} \\ \mathtt{veille\ feu\ obj?}\ \ Cp_Tach \\ \mathtt{eng\ tach_t-1} \end{array} \right) \mathrm{P}\left(\mathtt{Base} \;\middle|\; \begin{array}{l} \mathtt{px0..px7} \\ \mathtt{lm0..lm7}\ \ Cp_DétectBase \end{array} \right) \right) \tag{10.37}$$

par le programme suivant :

- Choisir pour la variable `Base` une valeur, `base`, tirée suivant la distribution :

$$\mathrm{P}\left(\mathtt{Base} \mid \mathtt{px0..px7\ lm0..lm7}\ \ Cp_DétectBase\right). \tag{10.38}$$

- Utiliser la valeur base et choisir pour la variable `Tach` une valeur `tach` tirée suivant la distribution :

$$\mathrm{P}\left(\mathtt{Tach} \mid \mathtt{base\ veille\ feu\ obj?\ eng\ tach_t-1}\ \ Cp_Tach\right). \tag{10.39}$$

Globalement, en appliquant une décision pour chaque somme de (10.35), le robot peut être commandé par le programme suivant :

- Décider si on est à la base en choisissant pour la variable `Base` la valeur `base` tirée suivant la distribution (10.38) ;
- Décider de la tâche à accomplir en choisissant pour la variable `Tach` la valeur `tach` suivant la distribution (10.39) ;
- Décider du type de déplacement à adopter en choisissant pour la variable Td la valeur `td` tirée suivant la distribution :

$$\mathrm{P}\left(\mathtt{Td} \mid \mathtt{tach\ td_t-1\ tempo\ tour}\ Cp_TypDepl\right). \tag{10.40}$$

- Décider de la position de la source lumineuse en choisissant pour la variable `ThetaL` la valeur `thetaL` tirée suivant la distribution :

$$\mathrm{P}\left(\mathtt{ThetaL} \mid \mathtt{lm0..lm7}\ Cp_SL\right). \tag{10.41}$$

- Décider de la valeur `h` de la variable de combinaison H en tirant suivant la distribution :

$$\mathrm{P}\left(\mathtt{H} \mid \mathtt{prox}\ Cp_PhotoEvit\right). \tag{10.42}$$

- Finalement, décider des ordres à envoyer aux moteurs du Khepera en tirant les valeurs de `Vrot` et de `Vtrans` suivant la distribution :

$$P(\texttt{Vrot Vtrans} \mid {}_{\texttt{h td thetaL dir prox dirG proxG vtrans_c}}\, Cp_Surveil). \tag{10.43}$$

La Figure 10-2 illustre l'application de décision pour chaque variable intermédiaire. Nous pouvons voir ainsi que le programme permettant de décider des valeurs des variables `Vrot` et `Vtrans` peut se traduire en six programmes indépendants exécutés successivement.

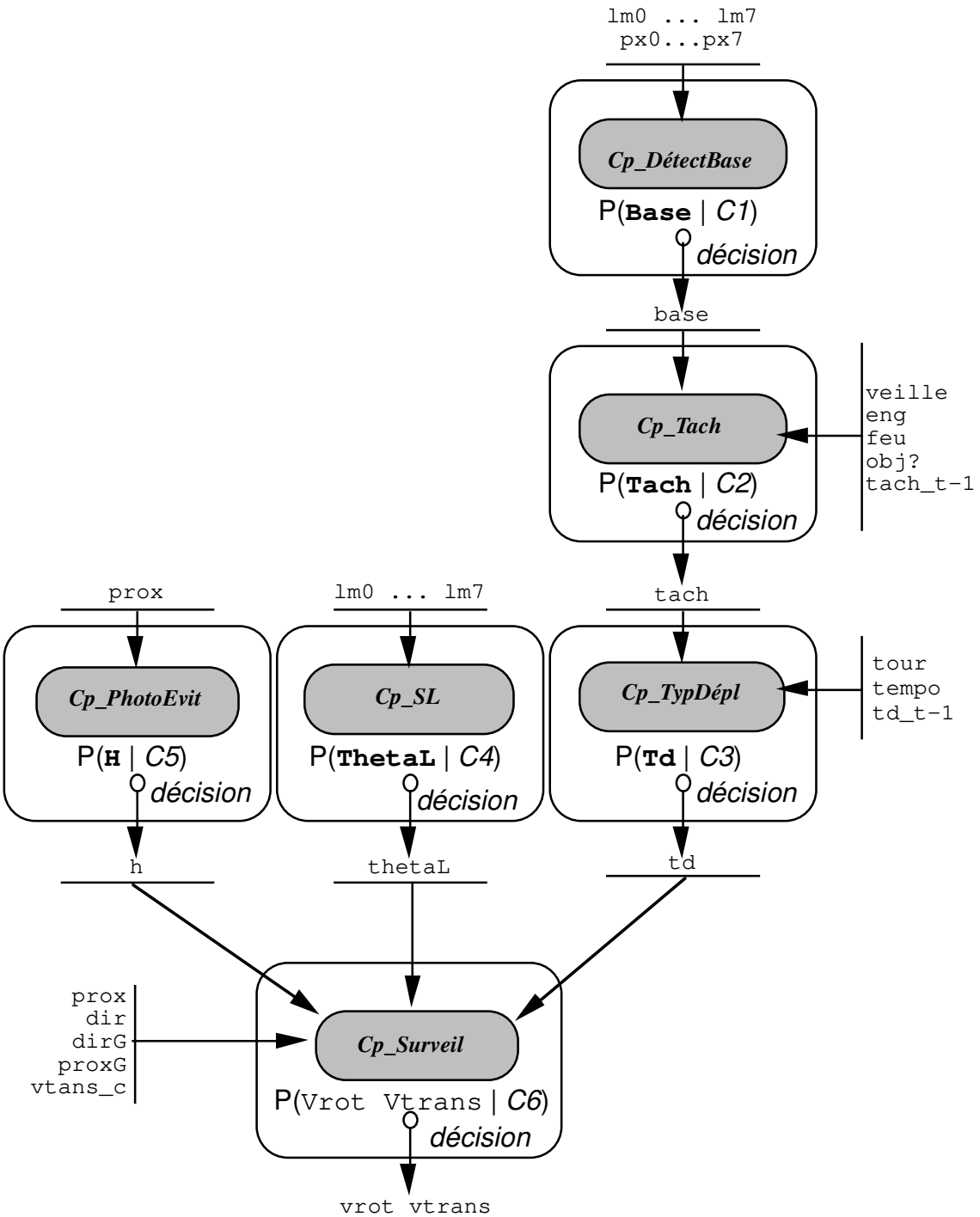


Figure 10-2 — Structure de la question permettant de calculer Vrot et Vtrans.
Le calcul est obtenu en effectuant une décision sur chaque variable intermédiaire. Cette figure illustre l'enchaînement hiérarchique des différentes questions intermédiaires, chacune correspondant cette fois ci à un programme particulier.

Toutes les solutions intermédiaires entre la propagation intégrale des incertitudes (Figure 10-1) et la décision pour toutes les variables intermédiaires (Figure 10-2) peuvent être

envisagées. La solution retenue pour cette expérience a consisté à décider des valeurs de `Base`, `Tach`, `Td` et `ThetaL` mais à propager l'incertitude sur `H` (pas de décision).

10.5.2 Résultat

Le résultat obtenu est très satisfaisant. Le Khepera a rempli sa tâche de « veilleur de nuit » des centaines de fois dans une très grande variété d'environnements et de conditions. Par exemple, cette expérience a tourné 12 heures par jour, sans interruption, durant 3 jours consécutifs dans le cadre de l'exposition la « Science en Fête ».

Le film suivant (Figure 10-3) montre le Khepera lors de l'une de ces expériences. On y voit successivement :

- Le Khepera en train de reconnaître un objet.
- Le Khepera interrompt la reconnaissance d'objet au moment de la détection d'un possible incendie (matérialisé par une bougie).
- Le Khepera vérifie qu'il s'agit bien d'une flamme en soufflant dessus.
- Le Khepera se rapproche tout en soufflant et réussit à éteindre l'incendie.
- Le Khepera repart en patrouille. Il s'arrête pour vérifier s'il détecte du mouvement dans son champ visuel.
- Le Khepera reçoit l'ordre de mettre fin à la patrouille et retourne à sa base.

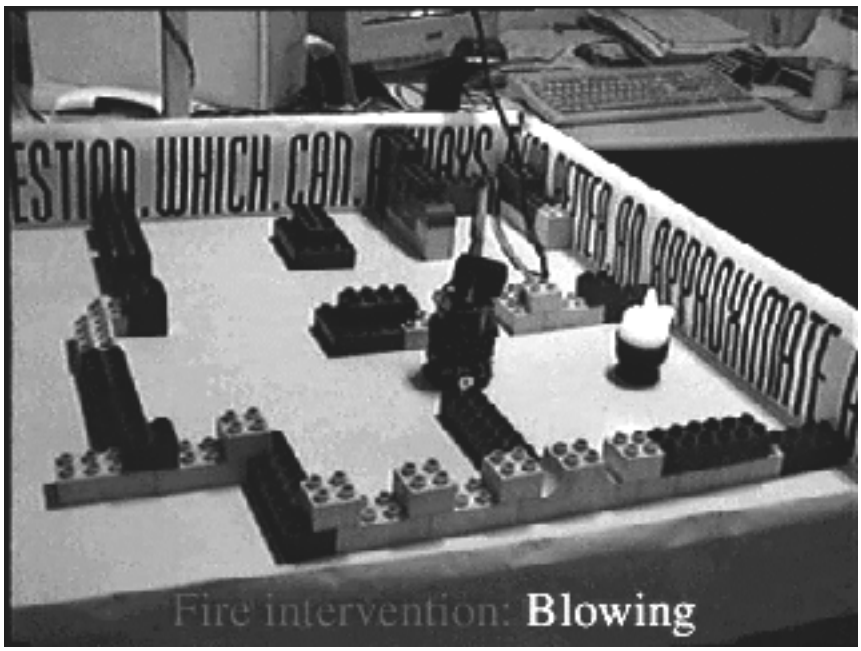


Figure 10-3 — Tâche de surveillance.
Au cours de cette séquence ; le Khepera débute par la reconnaissance d'un objet ; la détection d'un éventuel incendie interrompt la reconnaissance ; après la vérification qu'il s'agit bien d'un incendie, le Khepera éxécute le « programme pompier » avec succès ; puis repart en patrouille ; finalement, suite à un ordre de fin de patrouille, le Khepera rentre à sa base. (film : Film—Chp10.mov ; durée : 2 minutes 29 secondes).

10.6 Enseignements et discussion

Cette expérience a prouvé que l'intégration en une seule et même description de toutes les descriptions était possible et aisée.

Dans cette dernière expérience, nous pouvons mettre en évidence trois aspects de la notion de modularité : la structure déclarative, la structure procédurale et la structure opérationnelle.

- Structure déclarative : La spécification de l'interdépendance des modules permet de décrire une architecture structurelle. Dans le cadre de la tâche de surveillance, nous avons sélectionné différentes descriptions formant les briques de base de notre programme puis nous avons spécifié par l'expression d'indépendances conditionnelles comment ces modules se connectaient entre eux. C'est ce que nous appelons la structure déclarative.
- Structure procédurale : Cette structure est le résultat d'une requête particulière posée à la structure déclarative. La connaissance de certaines variables permet de faire apparaître des simplifications qui sont le fruit d'hypothèses d'indépendances conditionnelles. Cette structure est donc fortement dépendante de la requête posée. Cette structure se traduit par l'imbrication de sommes (marginalisation sur des variables inconnues dans la requête) et de produits de distributions, propageant l'incertitude au travers des différents modules, partant des variables connues aux variables recherchées. Cette propagation s'exprime sous la forme de requêtes particulières posées aux descriptions élémentaires.
- Structure opérationnelle : Finalement, afin d'optimiser les calculs, il est généralement possible de simplifier la structure procédurale en remplaçant certaines des sommes par des décisions. Le prix à payer pour cette optimisation des calculs est une perte d'information dans la mesure où chaque décision « fige » le raisonnement au lieu de propager l'incertitude. Cette dernière structure s'appelle structure opérationnelle.

Nous montrons que notre approche reste compatible avec la notion de modularité qui consiste à diviser un problème en sous-problèmes. Toutefois, ici, cette décomposition ne repose pas sur une indépendance forte entre les niveaux. La dépendance maintenue permet de propager l'incertitude entre les différents niveaux d'analyse d'un problème. Cette propagation peut être « ignorée » en effectuant une décision sur les variables intermédiaires. Une architecture de programmation semble finalement contingente à une question donnée (structure procédurale) et peut-être découpée en niveaux indépendants en jouant sur les décisions (structure opératoire). Ainsi, nous apportons un éclairage nouveau sur la notion d'architecture ; pour nous, l'architecture d'un programme traduit des choix de décision rendant « indépendant » les différents modules, c'est-à-dire ne propageant pas les incertitudes.

Chapitre 11

Synthèse

« Ce que nous appelons le hasard n'est, et ne peut être,
que la cause ignorée d'un effet connu. »

Voltaire, Dictionnaire philosophique.

Dans tous les domaines abordés par l'intelligence artificielle, les limites des *approches symboliques* deviennent de plus en plus manifestes. De nouvelles approches se développent, utilisant d'autres outils théoriques ou techniques. Elles sont souvent issues de recherches menées dans d'autres domaines que celui de l'intelligence artificielle, comme la biologie ou la physique. On a vu ainsi se développer des techniques d'optimisation basées sur la physique du solide (technique du recuit simulé [Kirkpatrick83]) ou sur la biologie (techniques de recherche stochastique comme les algorithmes génétiques [Holland75], [Goldberg89]) et des architectures de calcul basées sur la neurobiologie ([Rumelhart88], [Hertz91], [Haykin94], [Bishop95]).

Toutes ces approches se caractérisent par une remise en cause, plus ou moins profonde, des présupposés de l'intelligence artificielle telle qu'elle a été définie à ses débuts. Notre travail participe à cette remise en cause dans le cadre de la robotique. Dans la première partie de ce chapitre, nous situerons notre approche par rapport à deux autres : l'approche symbolique et l'approche constructiviste. Ensuite nous résumerons les principes de notre méthode et en donnerons ses principaux avantages. La dernière partie sera consacrée à de possibles suites pour ce travail et à notre conclusion.

11.1 Situation

Dans ce paragraphe, nous présentons les travaux en robotique en distinguant deux courants : l'approche symbolique et l'approche constructiviste. En pratique, cette distinction n'est évidemment pas aussi prononcée. Néanmoins, elle permet de fournir un éclairage sur la notion de représentation et comment celle-ci est abordée en robotique.

11.1.1 L'approche symbolique

Une caricature de l'*approche symbolique* en robotique est proposée Figure 11-1.

La démarche majoritairement suivie en robotique est la suivante. Le programmeur du robot a une conception (un modèle) abstraite de l'environnement. Conception en termes géométriques car il est capable de décrire la forme des objets et des lieux, en termes analytiques car il est capable de donner les lois de la physique qui régissent cet environnement et en termes logiques et ensemblistes car il peut nommer les objets et les classer suivant leurs caractéristiques pertinentes.

Le programmeur utilise cette formalisation pour programmer le robot. Ses programmes utilisent ces notions géométriques, fonctionnelles et logiques. Ainsi il « impose », au robot, sa propre conception de l'environnement.

Les difficultés de cette démarche apparaissent lorsque le robot doit faire le lien entre les signaux de bas niveau qu'il reçoit de ses capteurs et qu'il envoie à ses actionneurs et les concepts abstraits employés par le programmeur.

L'origine fondamentale de ces difficultés est, d'après nous, l'irréductible INCOMPLÉTUDE des modèles employés.

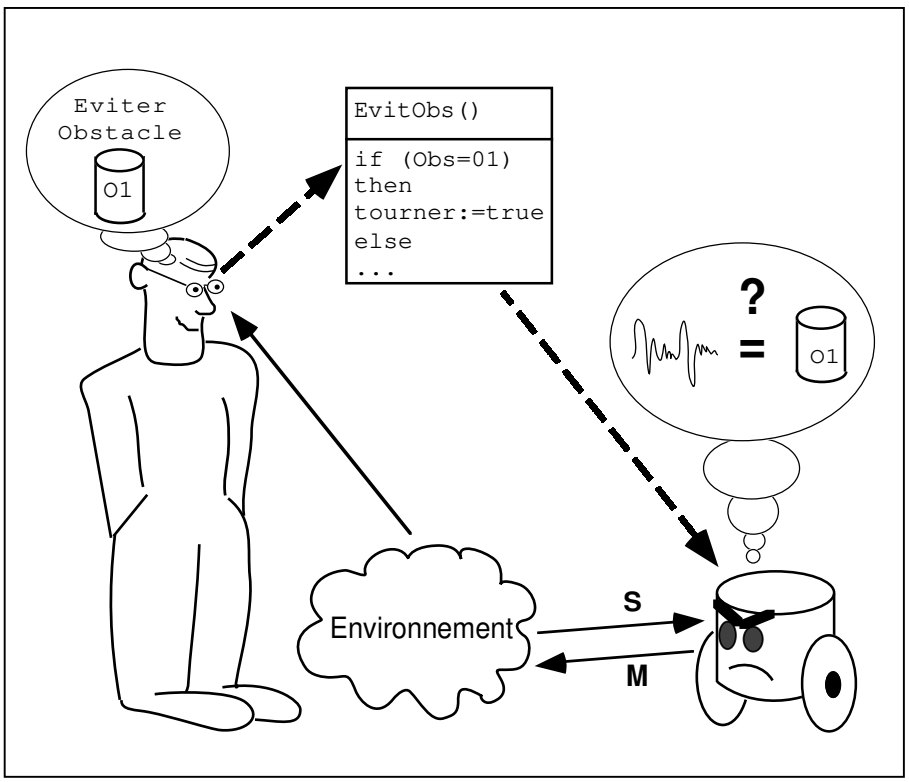


Figure 11-1 — L'approche symbolique en robotique.

En effet, quelle que soit la qualité du modèle fourni par le programmeur, il existe toujours des facteurs influençant le phénomène physique (l'interaction robot - environnement) qui n'ont pas été pris en compte. Les physiciens appellent ces facteurs des « variables cachées ». Ces variables cachées « brouillent » les données sensori-motrices. Le diagnostic souvent porté sur ce genre de situation consiste à dire que les données sensori-motrices sont « bruitées », voire quelquefois « aberrantes ». Étrange retournement de causalité qui semble considérer le modèle comme exact et imputer au monde physique quelque tare inconnue.

Une première possibilité est d'essayer d'améliorer le modèle pour qu'il prenne en compte la variable cachée gênante. Cependant, quoi qu'on fasse, un modèle ne peut jamais être complet. L'incomplétude est fondamentalement irréductible. Il existe toujours des facteurs non pris en compte qui, dans certain cas, peuvent soudain prendre une grande importance et occasionner de graves disfonctionnements.

La parade habituelle pour pallier à cette difficulté consiste à contraindre l'environnement dans lequel évolue le robot et les fonctions qu'il doit remplir. Le roboticien cherche les causes de ces « bruits » et modifie soit le robot, soit l'environnement, afin de supprimer ces « défauts ». Défauts signifiant ici que le monde physique a le mauvais goût de ne pas vouloir correspondre au modèle mathématique supposé le décrire ! L'environnement est donc contraint afin de correspondre au modèle mathématique qu'on en a. Cette démarche est légitime d'un point de vue ingénierie. Dans le cadre de développement d'applications robotiques industrielles, l'ingénieur se permet de contrôler étroitement l'environnement et de spécifier précisément les tâches que doit accomplir le robot. C'est cela qui assure la pertinence du modèle employé et le bon fonctionnement de ces applications.

Contrôler et contraindre l'environnement n'est cependant pas toujours possible. Dans cette thèse, nous nous sommes justement intéressés à la robotique en « environnement naturel », c'est-à-dire dans des environnements donnés au robot indépendamment de sa propre conception et ne pouvant être que partiellement décrits.

11.1.2 L'approche constructiviste

Une première voie dans ce sens, appelée *approche constructiviste*, consiste à essayer d'inverser la démarche qui voit le programmeur « imposer » son modèle au robot. Les aspects symboliques et « top-down » sont remplacés par une approche basée sur l'interaction avec l'environnement et fondamentalement « bottom-up ».

L'idée est de mettre le robot en *situation* pour qu'il construise ses propres représentations internes de son interaction avec l'environnement (définies dans ses termes sensoriels et moteurs). Le robot modifie ses représentations en tenant compte de son expérience. Il propose ensuite ces représentations au programmeur en vu d'un développement incrémental du robot. C'est au programmeur de faire le lien entre les représentations du robot et les siennes. En inversant ainsi la démarche, le problème est maintenant à la charge du programmeur et non plus à celle du robot (voir Figure 11-2).

Dans cette approche, une grande importance est donnée aux capteurs et aux actionneurs, ainsi qu'au traitement pertinent des informations. Elle a le mérite de s'attaquer aux problèmes de bas niveaux en montrant des résultats sur des robots réels et des tâches non triviales. Les systèmes robotiques proposés sont généralement réactifs et modulaires. Leur rapidité de réponse leur offre la capacité d'évoluer dans des environnements très dynamiques et, par la parallélisation et l'indépendance relative des modules, on leur accorde souvent une certaine robustesse. Leur fonctionnement repose sur un traitement adéquat des informations provenant du monde et nécessite des capteurs spécifiques à la tâche souhaitée.

Comme la caricature de cette approche le suggère (voir Figure 11-2), les expériences ainsi conduites ont montré que le programmeur n'arrivait généralement pas à résoudre de manière satisfaisante le lien entre symboles et représentations internes du robot. Le problème principal de l'utilisation de symbole critiqué par cette approche n'est pas résolu. En refusant l'aspect symbolique, le programmeur se confronte à la difficulté d'interpréter les représentations internes du robot, de les réutiliser ou de les faire évoluer vers une complexification « bottom-up ». Finalement, la possibilité de complexifier les systèmes de contrôle reste très limitée.

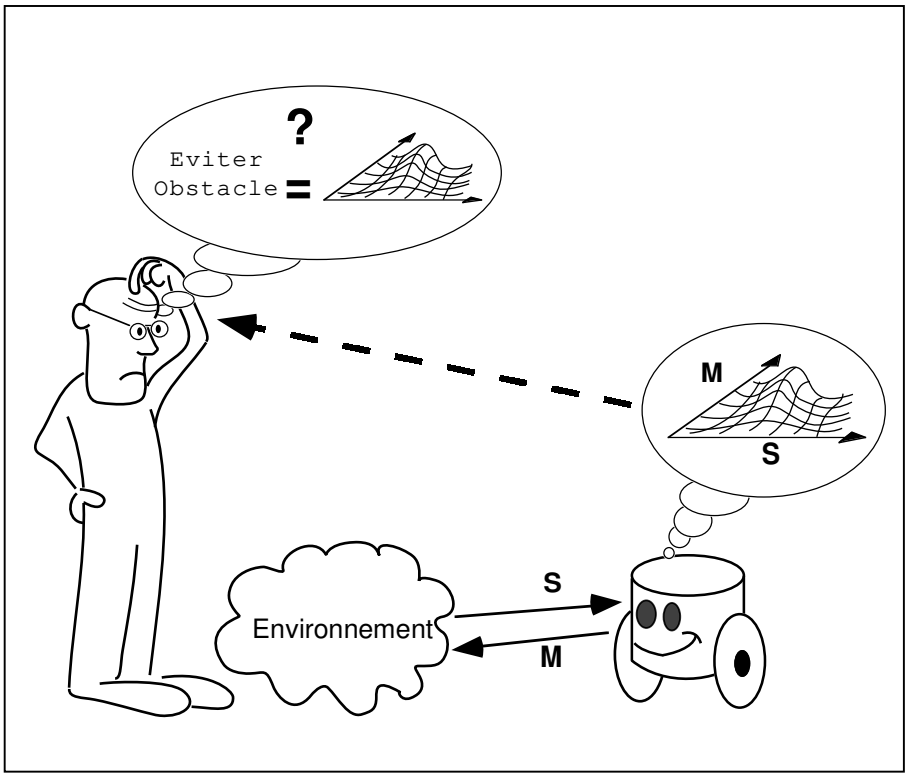


Figure 11-2 — L'approche constructiviste en robotique.

11.1.3 Notre approche

Ce que nous proposons dans cette thèse est une approche intermédiaire entre les deux précédentes fondée sur le paradigme bayésien. Notre spécificité est de se focaliser sur la notion de représentation interne et sur son statut inévitablement incomplet. Ainsi nous ne rejetons pas la notion de représentation, mais nous en modifions la nature et l'utilisation. L'idée de base est de confronter les connaissances préalables du programmeur à des données expérimentales pour construire des éléments de programmation que nous avons appelés *descriptions*. Notre démarche est ensuite de proposer ces *descriptions* comme outils de base pour la programmation, comme le suggère la Figure 11-3.

L'acquisition de représentations internes passe par une phase de modélisation et par une phase expérimentale où les données sont confrontées aux connaissances formalisées du concepteur. Cela permet de construire des structures de données représentant les dépendances entre variables sensorielles et variables motrices.

Le programmeur propose donc des connaissances préalables qui donnent au robot des indications sur ce qu'il peut s'attendre à observer lorsqu'il interagit avec son environnement. Ces connaissances préalables ne sont plus un modèle figé et rigide, prétendant à la

complétude comme dans l'approche symbolique. Ce sont des indices, des hypothèses qui pourront être confrontés à des données expérimentales en vue d'obtenir des descriptions.

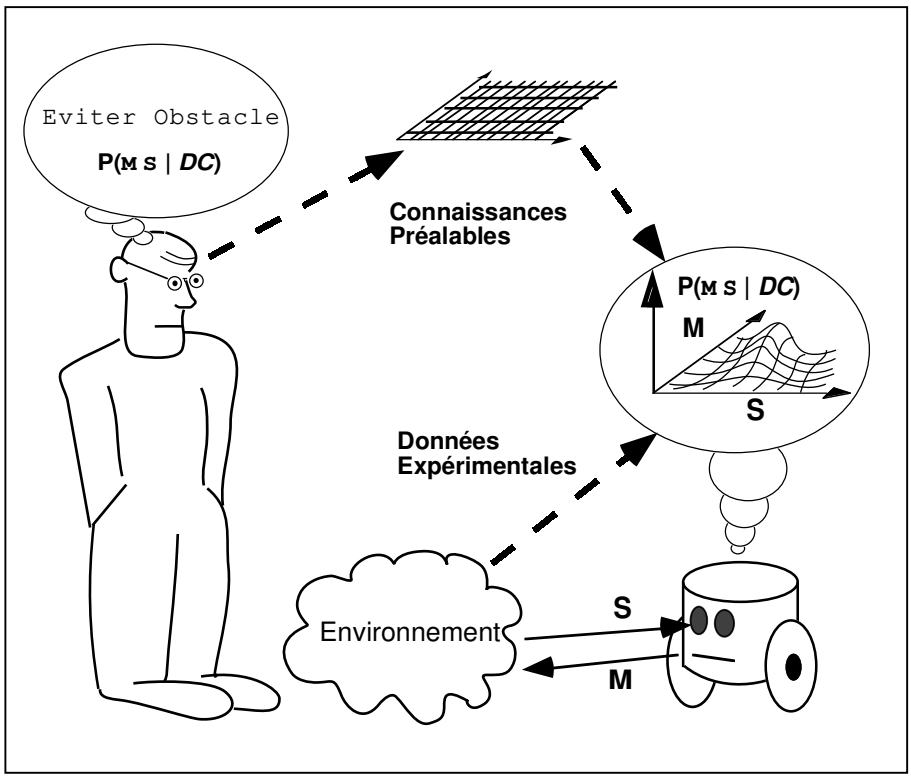


Figure 11-3 — L'approche suivie dans cette thèse

À la différence des approches classiques, les descriptions sont le résultat d'un apprentissage qui « fusionne » connaissances préalables et données. Par ce biais nous pouvons prendre en compte les spécificités du robot et l'influence des variables cachées. Notre méthode permet d'obtenir des représentations qui traduisent notre connaissance mais *aussi notre ignorance* des phénomènes d'interaction entre le robot et son environnement.

À la différence de la démarche constructiviste expliquée précédemment, le programmeur n'est plus totalement démuni pour comprendre les descriptions construites par le robot puisqu'il a fourni les hypothèses de base de ces descriptions avec les connaissances préalables. Il peut programmer le robot par le biais de questions probabilistes.

Les connaissances préalables peuvent être clairement et formellement explicitées en termes probabilistes. Il est possible de construire automatiquement les descriptions à partir des connaissances préalables et des données expérimentales. Il est aussi possible de tenir des raisonnements complexes avec ces descriptions en utilisant les règles d'inférence bayésienne. Tout l'intérêt de cette approche vient de ce qu'il existe un outil mathématique qui permet de la formaliser et de la rendre opérationnelle : les probabilités.

11.1.4 Notes bibliographiques

L'approche symbolique en robotique a hérité du *paradigme cognitiviste*, qui suppose que la cognition peut être définie comme une manipulation de représentations symboliques selon un ensemble de règles formelles, et être dissociée de son support physique (voir, par exemple, [Fodor76] et [Bickhard95]). Avec l'apparition de l'ordinateur, le paradigme cognitiviste est devenu la *métaphore computationelle* qui considère les processus mentaux comme des systèmes de traitement de l'information et le cerveau comme une sorte d'ordinateur. Une telle

définition suppose deux choses : d'une part l'existence d'une *interprétation* fixant sans ambiguïté le sens des symboles manipulés et, d'autre part, l'existence d'un niveau d'abstraction dans lequel les processus mentaux peuvent être considérés indépendamment du support physique et des situations qui les engendrent. Une telle perspective autorise une stratégie générale de résolution de problème, et a promu l'ordinateur comme machine cognitive universelle.

Dans ce cadre, la description d'un problème dans la machine est donnée par une représentation symbolique du monde. La validité de cette représentation repose sur une relation de correspondance entre l'état symbolique de la machine et l'état du monde donné par l'interprétation. Celle-ci présuppose une épistémologie objectiviste : implicitement on accepte l'existence d'un référentiel définissant un « état des choses dans le monde réel » pouvant être spécifié indépendamment d'une quelconque référence à un sujet. Outre la dimension épistémologique d'une telle position, se pose concrètement le problème de la mise en relation des symboles et des objets du monde réel au quels ils réfèrent (le « symbol grounding problem » cher à Harnad, [Harnad90]).

En informatique, cette mise en correspondance passe inévitablement par le jugement d'un agent extérieur. Cet agent (le programmeur ou l'utilisateur) encode l'information sous une forme qui peut être manipulée par les circuits électroniques. Ce faisant, cet agent fournit des informations sous forme symbolique que l'ordinateur implémente par exemple par la modification de différentes tensions électriques. L'ordinateur passe alors par différents états électroniques, pour restituer en retour d'autres symboles, image de l'état électronique final dans lequel il aboutit. Comme le fait remarquer Searle [Searle95], l'architecture électronique ne peut contenir ni syntaxe ni sémantique intrinsèque, « tout » est dans « l'œil » de l'utilisateur. C'est le concepteur (et l'utilisateur) qui décrit le monde sous forme symbolique, fournit cette description symbolique au système et en interprète les réponses. L'interaction avec le monde physique passe donc nécessairement par la médiation de l'utilisateur, qui est capable de passer de son registre cognitif à une formulation formelle (symbolique) et, en retour, d'interpréter dans son registre cognitif les résultats des manipulations formelles réalisées par l'ordinateur. Cela fait de l'ordinateur un concept universel indépendant de son substrat physique.

À la différence des ordinateurs, un robot est une entité particulière, spécifique : il n'en existe pas deux identiques. Un robot est supposé intervenir dans le monde physique sans l'intermédiaire d'un agent humain. Par sa nature, il est en médiation directe avec le monde physique. L'agent humain s'efface, il ne peut plus être présent à chaque étape de la boucle assimilant les informations en provenance du monde et produisant en retour les résultats du traitement de ces informations. La programmation de robot doit donc nécessairement prendre en compte cette spécificité.

Appliquée à la robotique, l'approche classique utilise un modèle de l'environnement, qu'il soit donné a priori ou obtenu à partir de mesures. En s'appuyant sur ce modèle, le calcul doit déterminer les actions du robot. C'est le domaine de la planification qui couvre deux problématiques, la génération du plan d'action (quelles actions réaliser) et l'ordonnancement (quand les réaliser), et qui s'inscrit dans une gamme plus large de résolution de problème (on trouve dans [Hendler91] une bonne compilation de ces travaux). Ces capacités de raisonnement de haut niveau dans la phase de planification rendent possible la gestion de missions complexes enchaînant plusieurs objectifs, dans des environnements multi-robots (voir par exemple [Alami96]). Dès lors, un des problèmes essentiels est de disposer d'un bon modèle de l'environnement. Parmi les modèles couramment utilisés en robotique classique, citons : les modèles CAO pour rendre compte des aspects géométriques du robot et de son

environnement ([Mazer91]) et les modèles dynamiques pour modéliser la réponse mécanique des robots à leurs systèmes de contrôle ([Borrelly98]). Le propre de ce type de représentations est d'être *exocentriques* pour le robot, a priori indépendantes de sa morphologie sensori-motrice. Ceci s'oppose aux représentations *égocentriques*, comme celles qui sont proposées par les approches constructivistes ou par l'approche préconisée dans cette thèse, qui se basent nécessairement sur les capacités sensorielles et motrices du robot.

L'approche classique aboutit à proposer des systèmes de contrôle fortement hiérarchisés, sur la base d'une décomposition du traitement de l'information. Ces architectures de contrôle reposent sur un ensemble de fonctions génériques, indépendantes et stables. Ces fonctions se succèdent, allant de l'entrée à la sortie : la perception, la modélisation, la planification et l'action. La perception consiste à évaluer l'état du robot dans le monde, distinguer la situation courante et fournir les informations décrivant cet état au module suivant. La modélisation se charge d'intégrer les informations fournies par la perception, de mettre à jour les connaissances du système décrites par une ou plusieurs représentations internes portant sur le monde externe et la tâche à accomplir. S'appuyant sur ces représentations internes, la planification recherche une suite d'actions permettant au robot de réaliser sa tâche. Enfin, le dernier module traite la suite d'actions fournie par la planification et contrôle en conséquence les effecteurs du système. Dans de telles architectures, les informations sont transmises de module en module. Différentes terminologies existent pour se référer aux étapes de cette boucle fonctionnelle de contrôle : Chris Malcolm évoque la boucle « *sense-think-act* » [Malcolm90], Rodney Brooks parle de boucle « *sense-model-plan-act* » [Brooks91c]. Cette décomposition en quatre modules peut ensuite se poursuivre, chaque problème étant décomposé en sous-problèmes, censés pouvoir être résolus séparément et indépendamment. On voit apparaître ici tous les niveaux habituellement distingués en robotique, que ce soit dans les architectures modulaires (niveau fonctionnel, niveau contrôle d'exécution, niveau décisionnel [Alami98]), ou dans les langages de programmation (langage de niveau effecteur, langage niveau tâche, par exemple LM, un langage de haut niveau conçu pour la commande de robots de manipulation et d'assemblage [Mazer84])

En pratique, l'indépendance des sous-problèmes n'est jamais acquise ; la modularité est plus un guide de conception qu'une technique applicable systématiquement. Dans sa thèse [Dedieu95], Eric Dedieu recense les difficultés de la robotique et procède à une analyse critique des limitations des différentes approches. La source conceptuelle des problèmes repose, selon lui, sur l'utilisation de représentations qu'il appelle « prédéfinies » (voir également [Stewart94]). Un modèle se caractérise et se justifie par la connaissance a priori de sa fonction. Modéliser un problème en termes prédéfinis, c'est faire le choix des termes exclusifs qui caractérisent ce problème. Tout en étant conscient des imperfections du modèle, le programmeur ne peut l'utiliser que sous l'hypothèse implicite qu'il est complet. Si les conditions de validité ne sont plus vérifiées, les prédictions du modèle seront considérées comme invalides. La notion même de fonctionnalité est alors abstraite et relative au jugement de l'observateur. L'emploi de termes prédéfinis fixe la « signification » d'un niveau de modélisation dans l'objectif de le rendre indépendant des autres niveaux et par la même de sa référence au monde concret.

Finalement, de telles approches ont longtemps conduit à la mise au point de programmes informatiques « déconnectés du monde », c'est-à-dire conçus indépendamment des spécificités physiques d'un robot particulier et de l'environnement dans lequel il est amené à évoluer.

Cette spécification des sous-domaines a abouti à l'apparition d'une dichotomie : les problèmes relevant de la cognition de haut niveau (« High-level Cognition ») accaparant en

grande partie l'attention et l'effort des chercheurs, et les problèmes du « bas niveau » (« Low-level Cognition »). L'intérêt qu'on y porte est plus récent et se retrouve dans une constellation d'approches alternatives telles que « Neo-connexionism », « Artificial Life », « Adaptative Systems » « Autonomous Robots », « Behavior-based AI », etc. Selon ces nouvelles approches, que John Stewart ([Stewart94]) propose de regrouper sous le terme d'approche constructiviste, la notion de représentation symbolique centrale ne semble plus nécessaire. En robotique, une des réactions prédominantes a été initiée par Rodney Brooks au MIT dans le milieu des années 80 ([Brooks86a], [Brooks86b]). La critique de Brooks repose sur le fait qu'en construisant un modèle de l'environnement, le système essaye d'obtenir une représentation d'une réalité extérieure objective.

La modélisation du monde sous la forme d'une description explicite des objets entourant le robot aboutit à imposer nécessairement une structure rigide sur un environnement qui, lui, est peu structuré. Le problème de déterminer quel aspect de l'environnement doit être modélisé et quelle interaction entre le robot et l'environnement est invariante ou non est une question difficile, connue sous le nom de « Frame Problem » [FramePb87]. Une autre facette de ce problème est de mettre en correspondance les mesures obtenues par les capteurs avec un modèle du monde inévitablement partiel. Le bruit des mesures introduit de l'incertitude.

D'après Brooks, le nœud du problème est donc de tenter de construire un modèle du monde. Il propose, tout simplement, de ne plus en construire ([Brooks91a], [Brooks91b], [Brooks91c]). Il rejoint en cela des propositions faites dans le domaine des sciences cognitives, où l'on parle d'un agent réactif dès lors « qu'il ne possède pas de représentation explicite du monde » ([Harel85]).

Dans l'approche réactive, on ne cherche pas à modéliser le monde réel, dynamique, mais plutôt à utiliser le monde lui-même comme modèle. Il s'agit de décomposer le comportement souhaité d'un robot en une collection de comportements simples, fonctionnant en parallèle. Un comportement simple est une action réflexe reliant directement des entrées sensorielles à des sorties motrices. Chacun des comportements réalise un couplage entre capteurs et effecteurs du robot. Les architectures de commande se basent sur la composition de plusieurs modules implémentant des comportements simples. Le comportement global du robot est le résultat de cette composition. Il *émerge* de l'interaction de ces comportements élémentaires entre eux, avec l'environnement ([Steels94], [Maes94]) ; et également entre différents robots ([Mataric96]).

Dans cette approche, la référence au vivant est fortement évoquée. Le robot est assimilé à une « créature ». Le programmeur peu à peu adopte le point de vue d'un éthologue observant le comportement de son « animat ». Une telle démarche a amené un rapprochement entre la robotique «Behavior-Based » et l'éthologie, et plus généralement la biologie (voir à ce propos [Schnepf91], [SAB91]). La robotique offre, en effet, un cadre d'expérimentation de théories du vivant (voir par exemple [Mataric90], [Gaussier95], [Webb95]), et réciproquement, elle peut hériter d'un « transfert de technologie » depuis la biologie vers les sciences de l'ingénieur (par exemple les travaux sur la vision de la mouche [Franceschini91], [Blumberg98], [Torralba99]). Toutefois, comme l'indique Raja Chatila dans [Chatila94], l'inspiration seule du vivant ne peut suffire. En cherchant à concevoir un système de contrôle pour un robot, nous devons aborder le problème d'un point de vue scientifique (en étudiant les principes de l'autonomie et de l'intelligence naturelle) mais également d'un point de vue de l'ingénieur (en cherchant des solutions spécifiques prenant en compte les objectifs et les contraintes matérielles).

Le principal reproche que l'on peut faire à l'encontre des architectures réactives vient de leur manque de capacité de raisonnement de haut niveau. Une forte motivation à l'application

de cette approche était l'opportunité de construire de manière incrémentale des robots et de les faire évoluer de façon à ce qu'ils aient des comportements de plus en plus complexes. Ce développement incrémental des robots s'est avéré extrêmement difficile à réaliser même si de nombreuses tentatives ont été faites dans ce sens ([Arkin89], [Maes89]). La difficulté centrale étant de créer les nouveaux capteurs spécifiques à un certain comportement désiré et d'insérer les nouveaux comportements dans l'interaction de ceux déjà implantés.

Les approches purement constructivistes ou purement symboliques représentent deux extrêmes que de nombreux travaux ont tenté de combiner. L'objectif de leur démarche est de conserver les capacités de raisonnement de haut niveau des approches symboliques, tout en assurant la robustesse des approches constructivistes (voir par exemple la thèse d'Eric Gauthier présentant un système de contrôle mêlant réseaux de neurones artificiels et logique floue [Gauthier99])

Le principe de la méthode que nous proposons dans cette thèse se fonde sur la distinction fondamentale entre le registre cognitif de l'observateur/concepteur humain d'une part et celui du robot d'autre part. Le registre cognitif humain inclut la modélisation formelle ainsi que la capacité d'interpréter les modèles, c'est-à-dire de relativiser leur utilisation. Du point de vue du robot, dans un environnement peu contrôlé, les valeurs sensorielles restent les seules sources d'information fiables. Nous cherchons donc à exprimer des *modèles* dans les termes du robot (*in fine* en termes sensoriels et moteurs). Pour cela, nous nous référons à un principe simple et intuitif déjà cité par Lloyd ([Lloyd88]) et Marr ([Marr82]) : toute dépendance effective observée entre les valeurs délivrées par deux capteurs physiquement et causalement indépendants est relative au contexte. Nous pensons de plus que, réciproquement, tout comportement pertinent du point de vue du robot doit se traduire par de telles dépendances observables ([Bessière94], [Dedieu94a], [Dedieu94b]). Dès lors, une représentation d'une dépendance sensori-motrice, la description, va fournir au robot le moyen d'identifier et de restituer un comportement, de juger que celui-ci est en relation avec l'environnement, qu'il a un « sens ».

À ce titre et au vu des expériences présentées dans ce travail, notre approche peut sembler proche du constructivisme. Toutefois, nous ne refusons pas l'utilisation de représentations internes, de modèles. En effet, une description est une forme de représentation de l'interaction du robot avec l'environnement, pouvant être plus ou moins abstraite. Afin que le robot soit en mesure d'observer des dépendances et de les traduire en une structure abstraite, il est nécessaire de lui fournir un ensemble d'informations : les connaissances préalables. Ces connaissances peuvent se traduire sous forme de modèles abstraits, complexes, nécessaires à la mise en évidence de relations pertinentes. Aussi riches soient ces connaissances, elles sont inévitablement incomplètes. Il faut alors accepter cette incomplétude et chercher un mode de représentation adapté à cet état de fait : les probabilités.

11.2 Contributions

Nous pouvons maintenant présenter la synthèse de notre approche et en donner les principaux avantages.

11.2.1 Fondements théoriques

Revenons brièvement sur les fondements théoriques des travaux présentés dans cette thèse, schématisés sur la Figure 11-4.

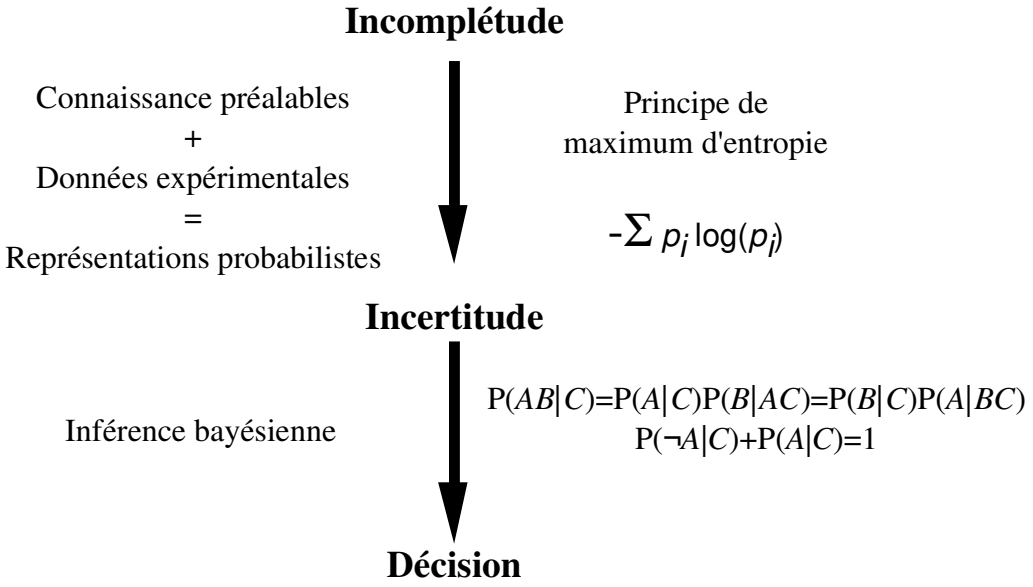


Figure 11-4 — Fondements théoriques : « de l'incomplétude à la décision »

Le point de départ est la reconnaissance de l'incomplétude comme la difficulté fondamentale à laquelle sont confrontés les systèmes sensori-moteurs.

La première étape consiste à transformer cette incomplétude en incertitude. Par l'apprentissage, à partir des connaissances préalables et des données expérimentales, des distributions de probabilité sont construites.

Le principe de « maximum d'entropie » est le fondement théorique de cette première étape. Étant donné un ensemble de connaissances préalables et un jeu de données, la distribution de probabilité qui représente le « mieux » ce couple est celle qui maximise l'entropie. L'entropie donne un sens précis, mathématique et quantifiable, à la « qualité » des distributions (voir [Jaynes82], [Kapur92], [Venditti98] et l'article [Bessière98b] inséré en annexe pour une justification du principe de maximum d'entropie).

Prenons deux exemples extrêmes pour mieux comprendre :

- Supposons qu'on cherche à modéliser un phénomène formel, sans variables cachées, pour lequel un modèle complet puisse être fourni. Si l'on prend comme connaissances préalables ce modèle et n'importe quel jeu d'observations de ce phénomène on obtiendra alors par apprentissage un ensemble de distributions de type Dirac qui traduiront les certitudes qu'on a sur ce phénomène. Les probabilités se réduisent alors à la logique et on retombe sur la première approche.
- Supposons à l'opposé que les connaissances préalables soient de très mauvaises hypothèses concernant le phénomène observé. L'apprentissage conduit alors à des distributions de probabilités très « plates » qui ne permettent de rien décider si ce n'est au hasard.

Les cas intéressants sont les cas intermédiaires, où les connaissances préalables bien qu'imparfaites expriment une information pertinente sur le phénomène. Les distributions obtenues par apprentissage ne sont alors ni des Diracs ni uniformes. On n'a pas de certitudes

sur le phénomène modélisé, mais on a des connaissances qui peuvent être utilisées pour décider au mieux des conduites à tenir.

La deuxième étape consiste à tenir des raisonnements avec les distributions de probabilité obtenues par la première étape pour finalement arriver à une décision motrice.

Il faut être capable de raisonner avec l'incertitude. L'inférence bayésienne et ses deux règles de base permettent ces calculs. Ces deux règles jouent pour le raisonnement bayésien le rôle que joue le principe de résolution ([Robinson65], [Robinson79] et [Robinson83]) pour le raisonnement logique. Ces raisonnements peuvent être aussi compliqués et subtils que ceux effectués avec le raisonnement logique habituel, comme l'ont montré de nombreux exemples vus dans les chapitres précédents.

Les fondements théoriques nécessaires se résument donc au principe de maximum d'entropie pour guider l'apprentissage et aux deux règles d'inférence pour tenir les raisonnements. Cette simplicité des fondements nous semble être une très grande force de l'approche proposée.

11.2.2 Méthode

La méthode que nous avons suivie pour programmer le Khepera découle directement de ces fondements théoriques. Rappelons là une dernière fois :

Pour programmer et réaliser une tâche robotique, la démarche à suivre est toujours la même et se décompose selon trois phases :

- *Phase de spécifications* où le programmeur donne les connaissances préalables par :
  - Le choix des variables pertinentes ;
  - Le choix de la décomposition de la distribution conjointe en distributions élémentaires ;
  - Le choix de la forme paramétrique associée à chacune de ces distributions élémentaires.
- *Phase d'identification* où les valeurs des paramètres libres des différentes distributions élémentaires sont déterminées. Ces valeurs sont soit le résultat d'une identification à partir de données expérimentales, soit fixées a priori, soit, éventuellement, le résultat de l'interrogation d'une autre description.
- *Phase d'utilisation* où la description est interrogée en posant des questions probabilistes. Par exemple, dans le cas du contrôle moteur, nous calculons la probabilité des valeurs de différentes variables motrices connaissant les valeurs de variables sensorielles. Nous obtenons ainsi une distribution de probabilité sur les valeurs de ces variables et appliquons une décision afin de déterminer les ordres à transmettre au robot. Généralement la décision consiste à tirer au hasard selon la distribution de probabilité obtenue.

Cette méthode présente une distinction entre deux aspects inhérents à la réalisation de tâches robotiques :

- Un aspect descriptif : l'explicitation des connaissances nécessaires pour décrire une dépendance (dimension déclarative) et l'adaptation de ces connaissances à une situation réelle via la sélection d'un jeu de données expérimentales (dimension empirique).

- Un aspect opératoire : l'exploitation des connaissances par la résolution systématique de questions probabilistes et l'application de méthodes de décision (dimension procédurale).

Notre méthode se résume à :

Une tâche robotique = une Description + une Question + une Décision

11.2.3 Avantages

Dans ce paragraphe, nous nous efforçons de recenser, commenter et discuter les avantages de la méthode de programmation des robots proposée dans cette thèse.

- *Capacité à traiter l'information incomplète et incertaine* : la motivation principale de ce travail vient du problème fondamental de la programmation des robots : l'irréductible incomplétude des modèles. La première qualité de l'approche proposée est donc de permettre de programmer les robots en tenant compte de cette incomplétude et de l'incertitude qui en découle. Ceci est rendu possible, comme il vient d'être rappelé au paragraphe 11.2.2, en procédant par étapes s'appuyant sur trois capacités de la méthode proposée :
  - Capacité à transformer l'incomplétude en incertitude par l'apprentissage, que démontrent les nombreux exemples d'identification paramétrique à partir de données expérimentales, comme dans les expériences d'apprentissage de comportements réactifs (Chapitre 4) ou dans les expériences de reconnaissance de situations (Chapitre 8). La reconnaissance des objets (section 8.2), par exemple, montre que par l'apprentissage, sur la base de connaissances préalables simples, on est capable de construire des distributions de probabilité suffisantes pour la classification. Les variables cachées ou non prises en compte qui peuvent influencer cette tâche sont pourtant nombreuses comme, par exemple, la couleur et la matière des objets reconnus, l'éclairage global de la pièce, la qualité approximative du comportement de suivi de contour ou la position dans laquelle ce suivi a commencé.
  - Capacité à raisonner en présence d'incertitudes que démontrent toutes les expériences faisant appel à de l'inférence comme, par exemple, la fusion d'information capteur (Chapitre 5) ou la reconnaissance d'objets (section 8.2). Le comportement de « veilleur de nuit » (Chapitre 10) montre la complexité des raisonnements tenus (42 variables, 12 descriptions, 4 niveaux hiérarchiques).
  - Capacité à décider en restituant l'incertitude : la stratégie de décision adoptée dans cette thèse consiste à tirer suivant la distribution obtenue à l'étape d'inférence précédente. Cette stratégie permet de restituer l'incertitude, les décisions étant quasi déterministes dans le cas de distributions très « piquées » et, inversement, très variables dans le cas de distributions très « plates ». Cette capacité à traduire l'incertitude dans les décisions prises est d'après nous très importante pour la qualité des comportements obtenus. Par exemple, en présence d'une très grande incertitude, le fait de tirer suivant une distribution quasi uniforme permet souvent au robot de se sortir d'un mauvais pas en retombant rapidement dans des situations, mieux connues, ou les choix sont plus tranchés (voir par exemple section 4.3.1 pour l'évitement d'obstacle frontal). La distribution correspondant à un comportement d'évitement d'obstacle peut être vue d'avantage comme la spécification de contraintes sur le mouvement plutôt que comme la spécification du mouvement lui–même. Cette distribution informe plus par les cas

qu'elle proscrit que par ceux qu'elle prescrit, elle élimine des choix plus qu'elle n'en impose. Cette faculté « proscriptive » plutôt que « prescriptive » des distributions est un point clef de la réussite de leur combinaison, la bonne décision étant, souvent, de chercher parmi les valeurs qui ne sont pas proscrites. Ce point est illustré par certaines expériences de combinaisons de comportements (Chapitre 6) et par l'expérience sur la fusion des informations capteurs (section 5.3). Dans cette dernière expérience, par exemple, la direction retenue pour la source lumineuse est souvent celle qu'aucun des capteurs ne considère comme totalement improbable.

- *Fondements théoriques simples et solides* : l'approche proposée s'appuie sur des fondements théoriques très simples et mathématiquement solides. C'est pour nous un avantage fondamental. Ces bases théoriques permettent de poser clairement et formellement les questions essentielles et d'envisager d'y répondre par le raisonnement mathématique. Ils permettent de mettre le raisonnement bayésien sur un pied d'égalité avec la logique et donc de l'envisager comme une alternative plausible en tant que modèle du raisonnement. Ils permettent d'analyser et de comprendre en détail les résultats expérimentaux, que ce soit des succès ou, plus enrichissant, des échecs. Ils guident les développements informatiques à effectuer, comme, par exemple, celui du moteur d'inférence bayésienne. Ils suggèrent les voies de recherche futures.
- *Méthode de programmation simple, systématique et générique* : l'approche proposée repose sur une méthode de programmation simple, systématique et générique. Simple, cette méthode peut être acquise et maîtrisée facilement et rapidement. Systématique, elle peut être appliquée avec rigueur et efficacité. Générique, elle s'applique non seulement à tous les problèmes de programmation des robots autonomes vus dans cette thèse mais aussi à de nombreux autres domaines comme, par exemple, celui de la CAO abordé dans la thèse de Kamel Mekhnacha [Mekhnacha99].
- *Homogénéité des représentations et des méthodes de résolution* : notre méthode repose sur une seule structure de données, la description, associée à deux règles de manipulation. Cette homogénéité permet des développements d'outils informatiques simples et génériques. Elle favorise aussi la clarté et la compréhension des programmes robotiques produits.
- *Obligation d'expliciter connaissances et hypothèses* : le choix de la description comme unique structure de données pour spécifier les problèmes robotiques et l'adoption d'une méthode systématique pour cette spécification obligent le programmeur à expliciter complètement et exhaustivement ses connaissances sur le problème. Tout ce qu'il y a à savoir sur un problème robotique se trouve dans sa description, c'est-à-dire dans l'union des connaissances préalables et des éventuelles données expérimentales. Aucune connaissance dépendant du problème n'est « cachée » dans les programmes permettant raisonnement et décision.

- Capacité à communiquer et comparer les modèles : il résulte de ce qui vient d'être présenté une très grande facilité à communiquer ou partager un modèle puisque la description résume tout. Les mêmes raisons permettent de discuter et comparer les différentes hypothèses d'un même problème avec une grande rigueur.
- Nouveau point de vue sur le dilemme « biais versus variance » et sur les problèmes de « généralisation » : La mise en avant des connaissances préalables, comme seul moyen d'exprimer ce que l'on sait a priori du problème traité, permet de trancher de nombreuses questions dans le débat « biais versus variance » [Geman92] ou dans celui sur la « généralisation ». De notre point de vue, tout biais, toute capacité à la généralisation ne peut provenir que des connaissances préalables. Un biais n'est que la

traduction du point de vue subjectif du programmeur sur les données qu'il analyse. À ce titre, le programmeur, et lui seul, a la responsabilité de ce biais. En contrepartie, il a la liberté d'imposer un biais aussi grand qu'il veut dans la mesure où celui-ci traduit ce qu'il pense effectivement, a priori, du phénomène observé. La situation est sensiblement la même pour la généralisation qui, d'après nous, n'est possible que dans la mesure où les règles pour généraliser ont été exprimées dans les connaissances préalables.

- *Pouvoir d'expression* : la description offre un très grand pouvoir d'expression aussi bien pour la spécification de modèles robotiques que pour leur interrogation.
  - Pouvoir de spécification : la variété des types de modèles spécifiés au cours de cette thèse démontre largement le pouvoir d'expression des descriptions. Rappelons qu'ainsi nous avons pu faire de la modélisation de capteurs, de la fusion d'informations capteur, de la programmation de comportements réactifs, de la combinaison de comportements réactifs, de la composition hiérarchique de descriptions probabilistes, de la reconnaissance et classification de situations, de la construction de séquences temporelles et, enfin, de la programmation de tâches complexes.
  - Pouvoir d'interrogation : rappelons aussi que la distribution conjointe permet de répondre à n'importe quelle question portant sur les variables apparaissant dans cette distribution (voir section 2.3.4). La variété des questions auxquelles on peut répondre par l'inférence bayésienne est donc très grande. Toutes les variables apparaissant dans la description conjointe jouent un rôle mathématique identique. Elles peuvent toutes, indifféremment, être soit considérées comme connues, soit inconnues, soit recherchées. La notion de « problème direct ou inverse » ([Tarentola87]) n'a plus lieu d'être. Cette caractéristique a été utilisée à plusieurs reprises dans les expériences de cette thèse où le modèle a été défini dans un sens pour être interrogé dans un autre. C'est notamment le cas pour la fusion d'information capteur (section 5.3), la reconnaissance de la base (section 8.1), la reconnaissance des objets (section 8.2) ou la programmation « inverse » décrite dans le Chapitre 9. La notion de « problème mal posé », c'est-à-dire admettant plusieurs solutions, très souvent lié à celle de problème inverse, n'a plus lieu d'être non plus. En effet, l'inférence bayésienne traite naturellement ces cas en donnant comme réponse des distributions réparties sur ces différentes solutions. Il en est ainsi, par exemple, pour le problème de la fusion d'information capteur (section 5.3), manifestement mal posé puisque plusieurs positions de la source lumineuse sont compatibles avec un même ensemble de lectures capteurs.
- *Capacité de développement incrémental des robots* : comme il a déjà été indiqué, les approches constructivistes se heurtent au développement incrémental des robots. L'approche que nous proposons, grâce à ses fondements théoriques solides et à sa méthodologie claire, est capable d'aller très loin dans ce sens. L'expérience finale (Chapitre 10), résultat de l'intégration de tous les développements précédents, est là pour le démontrer.
  - Capacité à combiner les descriptions : le premier outil de développement incrémental est donné par la capacité à combiner les descriptions, décrite au Chapitre 6. Cet outil permet de définir un nouveau comportement comme un mélange pondéré de comportements acquis précédemment.
  - Capacité à composer les descriptions : Le deuxième outil de développement incrémental est donné par la capacité à faire des compositions hiérarchiques, décrite au Chapitre 7

- Description = ressource : Plus généralement, la description, en tant que représentation interne d'un phénomène physique, peut être considérée, une fois acquise, comme une nouvelle ressource de programmation. Par exemple, une description peut fournir de nouvelles variables pouvant être utilisées dans d'autres descriptions. C'est le cas, par exemple, de la variable `ThetaL` rendue accessible par la description de fusion d'information capteurs (section 5.3) ou de la variable `Obj_id` rendue accessible par la reconnaissance d'objet (section 8.2). La reconnaissance des objets montre d'ailleurs une autre manière d'utiliser une description comme ressource de programmation, dans la mesure où le comportement de suivi de contour est l'instrument indispensable pour pouvoir calculer les quatre variables nécessaires à la reconnaissance des objets. Pour enrichir la gamme de possibilités d'un robot de nombreuses autres façons d'utiliser les descriptions sont envisageables. Plusieurs d'entre elles sont décrites dans la thèse d'Eric Dedieu ([Dedieu95]) où l'on trouvera, en plus, une analyse en profondeur de ces idées avec notamment la notion de « contingence ».

- *Une nouvelle approche de la modularité* : Notre méthode reste compatible avec le principe de modularité : décomposer un problème en sous-problèmes. Cependant, nous proposons d'aborder cette question d'une manière originale, à travers la composition hiérarchique des descriptions. Cette composition hiérarchique s'exprime sous la forme d'une description construite par la mise en relation de descriptions plus élémentaires en préservant la totalité des dépendances entre les différents modules. Nous avons mis en évidence dans la dernière expérience (Chapitre 10) trois facettes de cette modularité : la structure déclarative, la structure procédurale et la structure opérationnelle. Une architecture décomposant une tâche en modules indépendants est le reflet, selon notre approche, du choix d'effectuer des décisions entre deux niveaux au lieu de propager l'incertitude. Cette analyse apporte un nouvel éclairage sur la notion d'architecture informatique pour la commande des robots.

11.3 Perspectives

La méthode que nous présentons propose un cadre rigoureux pour exprimer des connaissances incomplètes et incertaines, raisonner sur ces connaissances et appliquer des décisions sur le résultat obtenu. Nous avons illustré ces différents points dans le cadre d'expériences relativement simples et intuitives. Cette simplicité, voulue dans un esprit didactique, masque les points durs inévitables de notre approche. Nous présentons ici les principaux et proposons les directions à suivre pour les aborder. Nous finirons par les différentes perspectives qu'offre ce travail.

11.3.1 Difficultés

En nous appuyant sur notre conception d'une tâche robotique (description + question) nous sommes en mesure de distinguer tout d'abord deux difficultés : d'une part la construction de la description et, d'autre part, son utilisation.

La définition d'une description et son utilisation par le biais d'une question et d'une décision peut servir de base à une discussion épistémologique dont les répercussions dépassent largement le cadre de cette thèse. Disons que l'élaboration d'une description est pour nous proche d'un problème de modélisation alors que son utilisation par l'inférence et la décision revêt plutôt un caractère technique (problème d'inversion) et architectural (temps de calcul).

11.3.1.1 Le problème de la spécification

Nous avons vu que nous pouvions scinder l'élaboration d'une description en trois points distincts : le choix des variables, la structure de dépendance et l'identification des paramètres. Ces choix ne sont pas indépendants et sont liés *in fine* à la tâche à accomplir. La mise au point d'une « bonne description » reste pour nous l'élément essentiel de la problématique de la programmation des robots et, pour l'instant, reste l'apanage du programmeur. Nous verrons que cette catégorisation des étapes de la modélisation peut offrir une brèche qui permet d'envisager une certaine automatisation et par cela un pas (timide) vers l'autonomie. Mais pour l'instant, examinons pour chacun de ces points quelles peuvent être les difficultés.

- Le choix des variables : Le choix des variables est un problème difficile et incontournable. Dans le jargon de l'équipe Laplace, nous nous accordons à nommer cette première difficulté « le problème des pré-traitements ». Un pré-traitement peut être vu comme le résultat de l'application de traitements spécifiques sur les variables de base afin de rendre l'information plus *compacte* et plus *pertinente*, c'est-à-dire mieux *adaptée* à la tâche que l'on souhaite réaliser. De plus, et c'est bien souvent le cas, à l'aide d'un pré-traitement nous cherchons à obtenir, à partir des signaux fournis par les capteurs, une information plus aisément *interprétable* du point de vue du programmeur. Enfin, nous pouvons souhaiter également obtenir par un pré-traitement une ressource *générique* pour programmer, c'est-à-dire suffisamment riche pour pouvoir être utilisée dans d'autres circonstances ou à d'autres fins. Ces différents objectifs sont rarement compatibles entre eux et, en pratique, aboutissent aux choix et à la mise en œuvre de différentes techniques pour les atteindre. Ces techniques sont nombreuses (échantillonnage, filtrage, dérivation, intégration, transformation logarithmique, analyse de données par regroupement, « clustering », analyse en composantes principales, quantification vectorielle, …) et recouvrent un large éventail de travaux allant de la robotique aux techniques d'analyse de données et de traitement du signal. Des techniques de mesure d'information (par exemple la mesure de cross-entropie) offrent un cadre intéressant pour quantifier l'adéquation relative d'un pré-traitement. Un pré-traitement peut être également obtenu en mettant en relation les informations fournies par les variables de base avec des modèles abstraits de l'environnement. Nous avons illustré modestement ce point dans l'exemple du modèle capteur (Chapitre 5) en montrant que l'approche bayésienne traite ce problème comme un problème d'inversion. Grâce au modèle direct (modèle du capteur) nous sommes en mesure de prédire les informations sensorielles (`Lmi`) connaissant les caractéristiques de « l'objet » modélisé. L'utilisation consiste à résoudre un problème d'inversion, c'est-à-dire, connaissant les valeurs des capteurs, prédire les valeurs caractérisant l'objet (`TheTaL` et `DistL`). Ainsi, nous voyons resurgir ici les deux approches décrites plus haut : « bottom-up » (partant des capteurs obtenir des pré-traitements pertinents) et « top-down » (partant d'une spécification abstraite de l'environnement, « l'ancrer » dans l'espace sensori-moteur du robot).
- Le choix de la structure de dépendance et des formes paramétriques associées est lui aussi délicat. Dans les réseaux bayésiens, des algorithmes existent qui cherchent le graphe du réseau au vu de données expérimentales (voir [Krause98], [Heckerman96]). Cette difficulté constitue une perspective intéressante pour des travaux futurs. Les données peuvent d'une part permettre de fixer les valeurs des distributions de probabilités, mais elles peuvent d'autre part servir à déterminer expérimentalement la structure de dépendance et les formes paramétriques. Dans le cadre de l'aide au diagnostic médical, Claudine Robert traite ces deux points par l'application du principe du maximum d'entropie comme un problème de choix d'observables (voir [Robert91]). En pratique, nous nous accordons la liberté de fixer à notre convenance les formes paramétriques ainsi

que la valeur des paramètres. Toutefois, il est clair que si nous y gagnons ainsi une plus grande richesse d'expressivité, nous sortons de la rigueur offerte par l'application de principes généraux tels que celui du maximum d'entropie. Dans ces cas, les solutions choisies deviennent plus difficilement justifiables voir simplement discutables.

- L'identification : enfin, l'obtention des valeurs des paramètres nécessite soit de les fixer a priori soit de fournir un jeu de données expérimentales. Nous avons appliqué ces deux alternatives. Fixer les valeurs des paramètres soulève le problème de quantifier une connaissance. Nous l'avons fait par exemple pour spécifier des niveaux de contraintes sur les commandes motrices (Chapitre 4), ou pour spécifier les séquences temporelles de tâches et de types de déplacement (Chapitre 9). L'influence des valeurs ainsi fixées et de leurs variabilités est difficile à juger a priori. Nous pensons que cette démarche permet de définir une forme brute à une distribution qui pourra par la suite être affinée, rectifiée par la confrontation avec un jeu de données expérimentales. Ceci nous mène au problème de l'obtention de ces données. Dans cette thèse, nous avons obtenu les informations expérimentales par télé-opération. La télé-opération permet d'obtenir une base représentative du comportement que peut donner un opérateur humain à un robot et de sa variabilité. La limite de ce procédé est dans l'habileté de l'opérateur et dans sa capacité à estimer les situations telles que les perçoit le robot. Dans le cas particulier du Khepera, la portée limitée de ses capteurs de proximité le rend souvent inapte à distinguer des situations sensorielles accessibles à l'opérateur. Celui-ci doit alors s'astreindre à télé-opérer un comportement donné dans la portée des capteurs du robot telle qu'il l'imagine. Mais cette capacité de se mettre « dans la peau » du robot est limitée. Malgré ces réserves, la télé-opération s'est révélée une tentative intéressante et ouvre des perspectives prometteuses que nous avons, au sein de l'équipe Laplace, commencé à explorer (voir section 11.3.2).

Toute approche visant à programmer un robot nécessite de faire des choix, poser des hypothèses, décider à quel niveau de finesse il est nécessaire de décomposer un problème. Le choix des variables pertinentes, l'expression de leurs interdépendances et le choix des modèles pour représenter ces interdépendances forment ce que nous appelons les connaissances préalables. Ces choix sont fondamentaux. L'approche que nous proposons cherche à rendre ces choix explicites. L'expression de ces connaissances illustre notre volonté de bien faire apparaître l'importance de ces choix, de ne pas l'occulter. Ceux-ci peuvent se révéler justifiés ou inadéquats. Quoi qu'il en soit ces choix sont fixés et déterminent un certain état de connaissance, base sur laquelle le robot va être en mesure de raisonner.

La description fournit un cadre rigoureux et très général afin d'expliciter les connaissances mises en jeu pour la réalisation d'une tâche robotique. La méthode que nous proposons ne dit rien sur le niveau de sophistication de ces connaissances. En particulier, elle ne dit pas que la notion de modèle symbolique est ou n'est pas nécessaire à la réalisation d'une tâche robotique donnée. Il est tout à fait clair pour nous que la réalisation d'une tâche peut nécessiter l'élaboration de modèles analytiques complexes même si nous n'en avons pas utilisé dans nos exemples. À ce niveau, l'apport de notre méthode est simplement de montrer que nous pouvons tenir compte de l'incomplétude des modèles en utilisant des données expérimentales.

Par cela nous voyons que nous laissons entier le problème de l'adéquation entre la conception d'une description et la tâche à réaliser.

11.3.1.2 Le problème de l'inférence

L'inférence bayésienne, nous l'avons illustré, est en mesure de répondre à toute question portant sur une partie des variables. Généralement, le calcul de la solution nécessite une marginalisation sur les variables ignorées (appartenant à l'ensemble $\mathbf{E}_i$). Cette marginalisation consiste à déterminer (ou estimer) une distribution de probabilité puis à rechercher les solutions les plus probables. Nous avons traité ce problème en énumérant tous les cas possibles. C'est une solution brutale et évidemment très coûteuse pour les espaces à grande dimension. Toutefois, dans les expériences traitées dans ce travail, les dimensions des espaces de marginalisation étaient suffisamment faibles pour qu'une énumération soit acceptable. Cette difficulté peut être abordée comme un problème d'optimisation ; en marginalisant, nous cherchons finalement les « pics » de probabilité dans un espace pouvant être très grand. Différentes techniques existent pour estimer la marginalisation et trouver les solutions les plus probables ; citons les algorithmes génétiques (solution retenue dans l'actuel moteur d'inférence), les algorithmes de type « recuit simulé », les méthodes de Monte-Carlo (solution retenue dans l'équipe Laplace dans la thèse de Kamel Mekhnacha [Mekhnacha99]). Dans certains cas, on peut envisager d´autres méthodes de résolutions (voir par exemple [Ruiz98] dans le cadre des mixtures de gaussiennes, [Jaakkola99] dans le cadre des modèles graphiques). Nous restons conscients de l'éventuel prix calculatoire nécessaire pour l'inférence. Cependant, nous parions que, comme le programmeur dispose de connaissances préalables et est en mesure de poser des hypothèses qui permettent de structurer la décomposition de la distribution conjointe, l'inférence devient ainsi possible. La tâche de surveillance, décrite au Chapitre 10, en est une illustration (la distribution globale porte sur 42 variables).

11.3.2 Perspectives à court terme

L'incertitude en robotique est un domaine largement exploré. Nous envisageons de nous intéresser plus particulièrement aux deux points suivants :

La prise en compte du temps, des séquences et de la planification : une tâche complexe a plusieurs dynamiques : les actions réflexes, les buts à court terme, les missions globales. Nous avons abordé en partie ce point dans le cadre de la tâche de surveillance. Un objectif sera de proposer des formalismes permettant de décrire et d'interpréter ces différentes dynamiques dans le cadre bayésien en nous appuyant sur des travaux déjà existants (voir par exemple les techniques basées sur des processus de Markov comme [Sutton98], POMDP [Cassandra94] ou les modèles formels d'architecture de contrôle, « the dual dynamics scheme » [Jaeger97]).

La possibilité d'ouverture et d'extension du système : l'un des avantages énoncés est de permettre un développement incrémental par composition de descriptions. Une des perspectives sera de s'intéresser aux limites pratiques de la composition en essayant de proposer des méthodes systématiques pour combiner, fusionner, composer des descriptions et d'envisager l'apprentissage dans des structures hiérarchisées. Il est nécessaire de faire le lien avec d'autres approches par exemple les travaux sur la fusion dans le cadre bayésien ([Piat97], [Bekerman92]) ou les travaux sur l'apprentissage dans des systèmes modulaires et hiérarchiques [Jordan95]). Au sein de l'équipe Laplace, un travail a déjà été engagé dans cette voie : apprentissage par téléo-pération de la combinaison de comportements (évitement d'obstacle et phototaxie) et de la séquence de tâches (voir [Diard99]). Hormis la télé-opération, il existe d'autres voies pour obtenir des données expérimentales (par exemple l'apprentissage par renforcement [Kaelbling96]) qu'il serait intéressant d'étudier dans le cadre de notre approche.

11.3.3 Perspectives à « plus long terme »

« Dans le domaine cognitif comme dans le domaine pratique, c'est en effet l'activité constructive qui paraît la caractériser le plus exactement. En ce sens, le propre de l'activité intellectuelle est de construire (et d'utiliser) des systèmes de représentations qui se substituent à la réalité perçue. Ces représentations peuvent être désignées comme des *modèles* de cette réalité. Et si l'on voulait exprimer ce fait sous une forme imagée, on aurait à dire que *l'intelligence est avant tout une machine à fabriquer des modèles.* »

P. Oléron, *Les activités intellectuelles*.

Il apparaît que nombre de nos contemporains qualifient aisément « d'intelligente » toute machine « manipulant » des modèles, *a fortiori* lorsque ceux-ci sont précisément d'hypothétiques modèles de l'intelligence.

Si le propre de l'activité intellectuelle est de construire des systèmes de représentation qui se substituent à la réalité perçue, c'est aussi dans leur utilisation qu'elle s'exprime finalement. Construire, inventer de nouveaux modèles, relève de capacités intellectuelles que nous nous accordons à associer à l'idée d'intelligence. Cette perspective nous éloigne d'une vision platonicienne centrée sur la relation objet connu - sujet et qui tend à maintenir le sujet dans un rôle passif, prenant conscience, accédant aux « Idées » …

Ainsi une des caractéristiques propre à l'intelligence réside non pas dans la faculté de prendre passivement connaissance d'un objet préexistant mais bien dans cette faculté à construire des objets nouveaux, à tirer de l'expérience de la réalité des substrats, des essences; c'est à dire à conceptualiser. C'est dans l'utilisation des résultats de ce processus de conceptualisation que nous redonnons « corps » aux concepts, en réincarnant, sous forme d'actions, les idées produites.

La robotique a un rôle fondamental à jouer au sein des sciences cognitives ([Vernant96], [Clark97]). Elle offre un cadre de réflexion et d'expérimentation dans lequel les problèmes de la construction de représentations internes, de leurs utilisations et de leurs validations sont renforcés et incontournables. Un robot doit interpréter les modèles qu'il utilise, il doit les mettre en relation avec l'environnement dans lequel il est plongé. In fine, il doit associer par un moyen ou un autre les valeurs de ses capteurs avec les abstractions qu'il manipule et en retour utiliser ces abstractions pour déterminer ses actions. Les modèles manipulés sont valides dans la mesure où ils permettent au robot d'agir dans son environnement de manière *efficace*. Dans quelles conditions un programme robotique peut-il s'effectuer de manière efficace et comment en juger ? Cette question est très difficile à résoudre.

L'efficacité et la validité d'un programme robotique sont indissociables de l'environnement, de la structure physique du robot, des objectifs attendus et de l'observateur qui finalement reste encore le seul juge. Ainsi, tant du point de vue de la conception des modèles que de l'évaluation de leurs adéquations, le rôle du programmeur reste incontournable. On peut cependant envisager de tester des conditions supposées suffisantes par le concepteur d'un programme. Il s'agira de mettre en place les procédures d'autotest permettant de valider certains a priori utilisés lors de la conception. Cette procédure visera à avertir l'utilisateur que les présupposés du concepteur ne sont pas respectés dans l'environnement courant.

Afin d'aborder l'autonomie en robotique, nous cherchons à poser le problème dans le cadre de la conception de robot autonome plutôt que dans le statut ou la définition d'un robot

autonome. Paradoxalement, c'est par l'étude et l'analyse du rôle du programmeur lors de la conception d'un robot que nous chercherons des voies pour la robotique autonome. De ce point de vue, notre objectif est de proposer des outils conceptuels, des méthodologies, des principes sur lesquels s'appuyer afin de proposer « à terme » des voies pour relâcher le rôle du concepteur et l'aider à transférer ses connaissances depuis son registre cognitif vers la machine.

À beaucoup plus long terme, nous pensons que l'inférence probabiliste mérite d'être évaluée comme paradigme computationnel alternatif au paradigme déductif et logique dominant en informatique et en sciences cognitives. Cela ouvre deux perspectives de recherche extrêmement ambitieuses et passionnantes :

- Au-delà du modèle de Turing, est-il possible de concevoir et de construire des machines probabilistes aptes à traiter de l'information incertaine et incomplète ? [4]
- Les connaissances, en très rapide progrès, portant sur le cerveau et le système nerveux des êtres vivants indiquent-elles une plausibilité biologique du traitement probabiliste de l'information ?[5]

[4] Cette idée n'est pas nouvelle, comme l'illustre la machine du type « Informon », mise au point par Uttley en 1969 [Uttley70]. Il s'agit d'un perceptron modifié de telle sorte que c'est la machine elle-même qui, calculant des probabilités conditionnelles dans les différents stimulis qui lui arrivent, fabrique en quelque sorte des *patterns* de fréquences d'après ces calculs, puis reconnaît plus ou moins ces *patterns* dans l'environnement.

[5] Voir par exemple les travaux de Weiss sur la vision humaine [Weiss98].

11.4 Conclusion

Nous avons présenté une approche originale pour la programmation des robots fondée sur une théorie de la représentation et du traitement des connaissances incertaines et incomplètes : « Probability as Logic ».

Pour cela nous avons utilisé un moteur d'inférence capable d'automatiser le raisonnement (déduction et induction) et l'apprentissage probabiliste. Nous avons intégré ce moteur dans un ensemble plus vaste de programmation de robot.

Nous avons utilisé ce système pour programmer une application de surveillance pour un robot mobile. Pour cela, nous avons utilisé des ressources génériques de programmation appelées descriptions. Nous avons montré comment obtenir et utiliser ses ressources dans un cadre systématique et unifié.

Les résultats les plus tangibles de cette thèse sont la définition d'une méthode de programmation de robot générique et son application expérimentale. La méthode ouvre de nombreuses perspectives dans le domaine de la modélisation et de la synthèse de systèmes autonomes. Elle permet de dégager formellement les étapes clés de la programmation des robots : description des connaissances préalables, identification, inférence et restitution ; et d'expliciter le rôle du concepteur.

Bibliographie

[Alami96] R. Alami ; *A Multi-Robot Cooperation Scheme Based on Incremental Plan-Merging* ; in R. Hirzinger & G. Giralt editors, Robotic Research : The Seventh International Symposium ; Berlin, Springer Verlag ;1996.

[Alami98] R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingrand ; *An Architecture for Autonomy* ; The International Journal of Robotics Research, Vol. 17, No. 4, pp. 315-337 ; April 1998.

[Arkin89] R. Arkin ; *Towards the unification of navigational planning and reactive control* ; Proceedings, American Association for Artificial Intelligence Spring Symposium on Robot Navigation, pp. 1-5 ; Stanford University, CA, March 1989.

[Bekerman92] M. Bekerman ; *A Bayes-Maximum Entropy method for multi-sensor data fusion* ; Proceedings of the 1992 IEEE International Conference on Robotics and Automation, Vol. 2, Nice, France, May 1992.

[Bessière87] P. Bessiere ; *LOLITA, Un langage de programmation logique intégré à LISP* ; Cahier technique ITMI ; 1987.

[Bessière94] P. Bessière, O. Lebeltel, E. Dedieu & E. Mazer ; *Representing Robot/Environment interactions using probabilities: the "Beam in the Bin" experiment* ; in P. Gaussier & J-D. Nicoud editors, Proceedings of the IEEE PerAc'94 Conference, Lausanne, Switzerland, pp. 158-169 ; 1994.

[Bessière98a] P. Bessière, E. Dedieu, O. Lebeltel, E. Mazer & K. Mekhnacha ; *Interprétation ou Description (I) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs* ; Intellectica, Revue de l'Association pour la Recherche Cognitive ; 1998/1-2, 26-27, pp 257-311 ; 1998.

[Bessière98b] P. Bessière, E. Dedieu, O. Lebeltel, E. Mazer & K. Mekhnacha ; *Interprétation ou Description (II) : Fondements mathématiques de l'approche F+D* ; Intellectica, Revue de l'Association pour la Recherche Cognitive ; 1998/1-2, 26-27, pp. 313-336 ; 1998.

[Bickhard95] M. H. Bickhard & L. Terveen ; *Foundation Issues in Artificial Intelligence and Cognitive Science ; Impasse and Solution* ; North-Holland, Elsevier Scientific ; 1995.

[Bishop95] C. M. Bishop ; *Neural Networks for Pattern Recognition* ; New York : Oxford University Press ; 1995.

[Blumberg98] B. Blumberg ; *Action Selection in Hamsterdam: Lessons from Ethology* ; Proceedings of the 3rd International Conference on the Simulation of Adaptive Behavior ; Brighton, MIT Press, August 1994.

[Borrelly98] J. J. Borrelly, E. Coste-Manière, B. Espiau & al, *The ORCCAD Architecture* ; The international Journal of Robotics Research, Vol. 17, No. 4, pp. 338-359 ; April 1998.

[Brooks86a] R. A. Brooks ; *Achieving artificial intelligence through building robots* ; AI MEMO 899, MIT ; 1986.

[Brooks86b] R. A. Brooks ; *A robust layered control system for a mobile robot* ; IEEE J. Robotics and Automation, 2(1), pp. 14-23 ; Avril 1986.

[Brooks91a] R. A. Brooks ; *Intelligence without representation*, Artificial Intelligence Journal, 47(13), pp. 139-160 ; 1991.

[Brooks91b] R. A. Brooks, *Challenge for complete creature architectures, From animals to animats* ; Proceedings of the First International Conference on Simulation of Adaptative Behavior, (Paris 1990) , J. A. Meyer & S. Wilson editors, MIT Press/Bradford Books, pp. 434-443 ; 1991.

[Brooks91c] R. A. Brooks ; *Intelligence without reason* ; Proceedings of the International Joint Conference on Artificial Intelligence, Sydney, Australia, Cambridge, MA ; MIT Press ; 1991.

[Cassandra94] A. R. Cassandra, L.P. Kaelbling & M. L. Littman ; *Acting optimally in partially observable stochastic domains* ; in Proceedings of the Twelfth National Conference on Artificial Intelligence, Seattle, WA ; 1994.

[Chatila94] R. Chatila ; *Control architectures for autonomous mobile robots* ; in P. Gaussier & J-D. Nicoud editors, Proceedings of the IEEE PerAc'94 Conference, Lausanne, Switzerland, pp. 254-265 ; 1994.

[Clark97] A. Clark, & R. Grush ; *Towards a Cognitive Robotics* ; Philosophy – Neuroscience – Psychology Technical Report ; report N° PNPTR-97-001 ; Washington University, St. Louis, MO ; 1997.

[Colmerauer86] A. Colmerauer ; *Theoretical model of PROLOG II* ; Logic Programmings and its Applications ; Ablex Publishing Corporation ; 1986.

[Cox46] R. T. Cox ; *Probability, Frequency and Reasonable Expectation* ; American Journal of Physics, 14(1), pp. 1-14, 1946.

[Cox61] R. T. Cox ; *The Algebra of Probable Inference* ; The John Hopkins Press, Baltimore ; 1961.

[Cox79] R. T. Cox ; *Of inference and inquiry, an essay in inductive logic* ; in The maximum entropy formalism, edited by Raphael D. Levine & Myron Tribus ; M.I.T. Press ; 1979.

[Dedieu94a] E. Dedieu & P. Bessière ; *La caractérisation sensorielle des comportements* ; NSI94 (Neurosciences et Sciences de l'Ingénieur); Chamonix ; 1994.

[Dedieu94b] E. Dedieu, E. Mazer & P. Bessière ; *La modélisation sensorielle: une approche originale de la notion de modélisation dans la programmation des robots*; Journées de Rochebrune « Autonomie et interactions fonctionnelles »; Mégève ; 1994.

[Dedieu95] E. Dedieu ; *La représentation contingente, vers une réconciliation des approches fonctionnelles et structurelles de la robotique autonome* ; Thèse de l'Institut National Polytechnique de Grenoble, spécialité « Informatique », Grenoble ; Septembre 1995.

[Diard99] J. Diard, *Apprentissage hiérarchique bayésien* ; rapport de DEA « Informatique, Systèmes et Communications » ; Université Joseph Fourier, Grenoble ; Juin 1999.

[Fodor76] J. A. Fodor ; *The Language of Thought* ; Harvester Press, Sussex ; 1976.

[FramePb87] *The Frame Problem in Artificial Intelligence* ; Proceedings of the 1987 Workshop, Morgan Kaufmann Publishers ; 1987.

[Franceschini91] N. Franceschini, J. M. Pichon & C. Blanes ; *Real time visu-motor control : from flies to robot* , in Proceedings of the IEEE Fith International Conference on Advanced Robotics 91 ICAR, pp. 931-935 ; June 1991.

[Gaussier95] P. Gaussier & S. Zrehen ; *Moving the Frontiers between Robotics and Biology* ; Robotics And Autonomous Systems (16)2-4, pp. 5-7 ; 1995.

[Gauthier99] E. Gauthier ; Utilisation de Réseaux de Neurones Artificiels pour la Commande d'un Véhicule Autonome ; Thèse de l'Institut National Polytechnique de Grenoble, spécialité : « Informatique, Systèmes et Communications », Grenoble ; Janvier 1999.

[Geman92] S. Geman, E. Bienenstock & R. Doursat ; *Neural Networks and the Bias/Variance Dilemma* ; Neural Computation 4, pp. 1-58, 1992.

[Goldberg89] D. E. Goldberg ; *Genetic Algorithms in Search, Optimization, and Machine Learning* ; Addison-Wesley ; Reading, MA ; 1989.

[Halpern99] J. Y. Halpern ; *A Counterexample to Theorems of Cox and Fine* ; Journal of Artificial Intelligence Research Vol. 10, pp. 67-85, 1999.

[Harnad90] S. Harnad. *The symbol grounding problem* ; Physica, D 42 : pp. 335-346, 1990.

[Harel85] D. Harel & A. Pnueli, *On the development of reactive systems* ; in K. R. Apt editor, Logics and Models of Concurrent Systems, Springer-Verlag, pp. 477-498 ; 1985.

[Haykin94] S. Haykin ; *Neural Networks : A Comprehensive Foundation* ; New York : Macmillan College Publishing ; 1994.

[Hendler91] J. Hendler, A. Tate & M. Drummond *; AI planning : Systems and Techniques* ; Proceedings of The Eighth National Conference on Artificial Intelligence AAAI'90, Morgan Kaufmann Publishers, Inc, pp. 61-77 ; 1991.

[Hertz91] J. Hertz, A. Krogh & R. G. Palmer ; *Introduction to the theory of neural computation* ; Addison-Wesley, Redwood City, CA, 1991.

[Holland75] J. H. Holland ; *Adaptation in Natural and Artificial Systems* ; University of Michigan Press ; 1975.

[Horswill93] I. Horswill ; *Specialization of perceptual processes*, PhD, MIT AI-Lab ; 1993.

[Jaakkola99] T. S. Jaakkola & M. I. Jordan ; *Variational Probabilistic Inference and the QMR-DT Network* ; Journal of Artificial Intelligence Research Vol 10, pp. 291-322 ; 1999.

[Jaeger97] H. Jaeger & T. Christaller ; *Dual Dynamics: Designing Behavior Systems for Autonomous Robots* ; In: S. Fujimura, M. Sugisaka, editors, Proceedings of the International Symposium on Artificial Life and Robotics (AROB '97), Beppu, Japan, pp. 76-79 ; Février 1997.

[Jaynes82] E. T. Jaynes ; *On the Rationale of Maximum-Entropy methods* ; Proceedings of the IEEE, Vol. 70, No. 9, pp. 939-952 ; Septembre 1982.

[Jaynes95] E. T. Jaynes ; *Probability theory – The Logic of science* ; inachevé ; Disponible à http://bayes.wustl.edu.

[Jordan95] M. I. Jordan, & R. A. Jacobs ; *Learning in modular and hierarchical systems* ; In M. Arbib, editor, The Handbook of Brain Theory and Neural Networks ; Cambridge, MA ; MIT Press, 1995.

[Jung97] D. Jung, K. K. Gupta ; *Octree-Based Hierarchical Distance Maps for Collision Detection* ; Journal of Robotic Systems ; Vol. 14, No. 11, p. 789-806 ; 1997.

[Kaelbling96] L. P. Kaelbling, M. L. Littman & A. W. Moore ; *Reinforcement Learning : A Survey* ; Journal of Artificial Intelligence Research Vol. 4, pp. 237-285 ; 1996.

[Kapur92] J. N. Kapur & H. K. Kesavan ; *Entropy optimization principles with applications* ; Academic Press Inc ; San Diego ; 1992.

[Kirkpatrick83] S. Kirkpatrick, C. D. Gelatt & M. P. Vecchi ; *Optimization by simulated annealing* ; Science 220 : pp. 671-680 ; 1983.

[K-Team94] K-Team, *Khepera User Manual Vers. 3.0* ; Documentation technique, K-Team, Lausanne ; 1994.

[Laumond90] J. P. Laumond, M. Taix & P. Jacobs ; *A Motion Planner for car-like robots based on a mixed global/local approach* ; Proceedings of IROS, Tsuchiura, Japan ; Juilliet 1990.

[Lauritzen88] S. Lauritzen & D. Spiegelhalter ; *Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion)* ; Journal of Royal Statistical Society B Vol. 50, No. 2, pp. 157-224 ; Janvier 1988.

[Lloyd88] D. Lloyd ; *Simple Minds* ; MIT Press ; BradFord Books ; 1988.

[MacKay96] D. G. C. MacKay ; *Introduction to Monte Carlo Methods* ; Proceedings of an Erice summer school, M. Jordan editor ; 1996.

[Maes89] P. Maes ; *How to Do the Right Thing* ; Connection Science Journal, Vol. 1, No. 3, pp. 291-323, 1989 ; Also MIT AI-Memo 1180 ; Decembre 1989.

[Maes94] P. Maes ; *Modeling Adaptative Autonomous Agents* ; Artificial Life Journal, Vol. 1, N° 1&2 ; MIT Press ; 1994.

[Malcolm90] C. Malcolm, T. Smithers & J. C. Hallam ; *An emerging paradigm in robot architecture*; In T. Kanade, F. C. A Groen. & L. O. Hertzberger editors ; *Intelligent Autonomous Systems*, Proceedings of an International Conference held in Amsterdam, pp. 545-564 ; 1989.

[Marr82] D. Marr ; *Vision* ; W.H. Freeman & Co ; 1982.

[Mataric90] M. J. Mataric ; *Navigating with a rat brain: a neurobiologically inspired model for robot spatial representation* ; In J-A. Meyer & S. Wilson, editors, Proceedings, From Animals to Animats 1, First International Conference on Simulation of Adaptive Behavior ; Cambridge, MA: MIT Press, pp. 169-175 ; 1990.

[Mataric96] M. J. Mataric ; *Learning in Multi-Robot Systems* ; in Adaptation and Learning in Multi-Agent Systems, Gerhard Weiss & Sandip Sen editors, Lecture Notes In Artificial Intelligence (LNAI), 1042, Springer-Verlag, pp.152-163 ; 1996.

[Mazer84] E. Mazer & J. F. Miribel ; *Le langage LM : Manuel de référence* ; CEPADUES éditions, 1984.

[Mazer91] E. Mazer & al.; *ACT a robot programming environment* ; IEEE International Conference on Robotics and Automation ; Sacramento, Avril 1991.

[Mekhnacha99] K. Mekhnacha ; *Méthode probabiliste bayesienne pour la prise en compte des incertitudes géométriques : Application à la CAO robotique* ; Thèse de l'Institut National Polytechnique de Grenoble, spécialité « Informatique » ; Grenoble ; Juillet 1999.

[Mondada93] F. Mondada, E. Franzi ; *Biologically inspired mobile robot control algorithms* ; Proceedings of the NRP23-Symposium on Artificial Intelligence and Robotics, Zürich ; 1993.

[Mondada94] F. Mondada, E. Franzi & P. Ienne ; *Mobile robot miniaturisation : a tool for investigation in control algorithms* ; Experimental Robotics III, Proceedings of the 3rd Symposium on Experimantal Robotics, Springer-Verlag, pp. 501-513 ; 1994.

[Neal93] R. M. Neal ; *Probabilistic inference using Marcov Chain Monte Carlo Methods* ; Technical Report CRG-TR-93-1 ; Dept. of Computer Science University of Toronto, 1993.

[Pearl91] J. Pearl ; *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference* ; Morhgan Kaufmann ; San Mateo, Ca ; 1988 (2nd edition 1991).

[Poggio84] T. Poggio ; *Low-level Vision as Inverse Optics* ; In Proceedings of Symposium on Computational Models of Hearing and Vision, M. Rauk (ed.), Academy of Sciences of the Estonian S.S.R., pp. 123-127 ; 1984.

[Robert91] C. Robert ; *Modèle statistique pour l'intelligence artificielle : l'exemple du diagnostic médical* ; Masson, Paris, France ; 1991.

[Robinson65] J. A. Robinson ; *A Machine Oriented Logic Based on the Resolution Principle* ; Jour. Assoc. Comput. Mach. ; Vol. 12 ; 1965.

[Robinson79] J. A. Robinson ; *Logic : Form and Function* ; North-Holland ; New York, USA ; 1979.

[Robinson83] J. A. Robinson & E. E. Silbert ; *LOGLISP : an alternative to PROLOG* ; Machine Intelligence ; Vol. 10 ; 1983.

[Ruiz98] A. Ruiz, P. E. Lopez-de-Teruel & M. C. Garrido ; *Probabilistic Inference from Arbitrary Uncertainty using Mixtures of Factorized Generalized Gaussians* ; Journal of Artificial Intelligence Research Vol. 9, pp. 167-217 ; 1998.

[Rumelhart88] D. E. Rumelhart, G. E. Hinton, & R. J. Williams ; *Learning internal representation by error propagation* ; In D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, editors, Parallel distributed processing : Explorations in the microstructure of cognition ; volume 1 : Foundations, pp. 318-362, MIT Press, Cambridge, MA., 1986.

[SAB91] J-A Meyer & S. W. Wilson editors ; *From Animals to Animats*: Proceedings of the First International Conference on Simulation of Adaptive Behavior ; Cambridge, MA: MIT Press ; 1991.

[Schnepf91] U. Schnepf ; *Robot Ethology : A proposal for research into Intelligent Autonmous Systems* , From animals to animats, Proceedings of the first international conference on simulation of adaptative behavior, (Paris 1990) , J-A. Meyer & S. W. Wilson editors ; MIT Press/Bradford Books, pp. 465-474 ; 1991.

[Searle95] J. R. Searle ; *The Critique of Cognitive Reason* ; in Readings in philosophy and cognitive science, A. I. Goldman (ed.) ; a Bradford Book ; MIT Press, Cambridge, Massachusetts ; pp. 833-847 ; 1995 (2nd edition 1995).

[Shachter86] R. Shachter ; *Evaluating Influence Diagrams* ; Operations Research Vol. 34, No. 6, pp. 871-882 ; 1986.

[Steels94] L. Steels ; *Emergent functionality of robot behavior through on-line evolution* ; In R. Brooks and P. Maes, Eds., Proceedings, Artificial Life IV, the Fourth International Workshop on the Synthesis and Simulation of Living Systems ; Cambridge, MA: MIT Press, pp. 8-14 ; 1994.

[Stewart94] J. Stewart ; *The implication for Understanding High-level Cognition of a Grounding in Elementary Adaptative Systems* ; in P. Gaussier & J-D. Nicoud editors, Proceedings of the IEEE PerAc'94 Conference, Lausanne, Switzerland, pp. 312-317 ; 1994.

[Sutton98] R. S. Sutton, D. Precup & S. Singh ; *Between MDPs and Semi-MDPs : Learning, Planning, and Representing Knowledge at Multiple Temporal Scales* ; Technical Report 98-74, Dept. of Computer Science, University of Massachusetts, Amherst, MA 01003 ; Avril 1998.

[Tarentola87] A. Tarentola ; *Inverse Problem Theroy : Methods for data fitting and model estimation* ; Elsevier ; New York, USA ; 1987.

[Torralba97] A. B.Torralba & J. Hérault ; *An efficient neuromorphic analog network for motion estimation* ; IEEE Transactions on Circuits and Systems-I : Special Issue on Bio-Inspired Processors and CNNs for Vision. Vol 46, No. 2 ; Février 1999.

[Venditti98] V. Venditti ; *Aspect du principe de maximum d'entropie en modélisation statistique* ; Thèse de l'Université Joseph Fourier ; discipline : Mathématiques Appliquées ; Grenoble ; Juillet 1998.

[Vernant96] D. Vernant ; *L'intelligence de la machine et sa capacité dialogique*, in Penser l'esprit : des sciences de la cognition à une philosophie cognitive ; D. Fisette & V. Rialle édts; Presses Universitaires de Grenoble ; 1996.

[Weiss98] Y. Weiss & E. H. Adelson ; *Slow and Smooth : a Bayesian theory for the combination of local motions signals in human vision* ; MIT AI Memo 1624 ; Février 1998.

[Yoshikawa91] Yoshikawa & al. ; *Identification of the Center of Friction from Pushing an Object by a Mobile Robot* ; Proceedings of the IEEE/RSJ International Conerence on Intelligent Robots and Systems IROS'91 ; pp. 449-454, 1991.

Annexe A

Cette annexe présente l'article « *Interprétation ou Description (I) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs* » parue, en 1998, dans Intellectica, revue de l'Association pour la Recherche Cognitive :

P. Bessière, E. Dedieu, O. Lebeltel, E. Mazer & K. Mekhnacha ; *Interprétation ou Description (I) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs* ; Intellectica, Revue de l'Association pour la Recherche Cognitive ; 1998/1-2, 26-27, pp 257-311 ; 1998.

Pierre Bessière
Eric Dedieu *Laboratoire LEIBNIZ - CNRS*
Olivier Lebeltel
Emmanuel Mazer
Kamel Mekhnacha

INTERPRÉTATION OU DESCRIPTION (I)

Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs

RÉSUMÉ : Jusqu'à quel point agir et percevoir supposent-ils de « comprendre » ou même plus simplement de se « représenter » le monde ? Telle est l'une des préoccupations fondamentales des recherches sur la sensori-motricité. Elle est au centre de nombreux débats en sciences cognitives. Cette question une fois formalisée, expurgée, simplifiée et traduite en termes mathématiques, nous amène à nous interroger tout au long de cet article, sur les liens qui peuvent exister entre les inférences formelles mécanisées informatiquement et leurs contreparties dans le monde physique où évolue un robot. Ainsi reformulée, la question centrale débattue devient : comment rendre effectives *les inférences formelles ?*

MOTS-CLÉS : science cognitive, intelligence artificielle, robotique, logique, inférence probabiliste, inférence bayésienne, entropie maximum, perception, action, décision

ABSTRACT : Is it necessary to « understand » or even to « represent » the world to perceive and act ? This is one of the fundamental issue in sensori-motor research. Numerous debates in cognitive sciences turn on this subject. After simplification, formalization and translation into mathematical terms, we reformulate this question as : what is the nature of the link between the automatic inferences of a computer and their intended counterparts in the physical world where a robot is to be « embodied » ?

KEY WORDS : cognitive science, artificial intelligence, robotics, logic, probabilistic inference, bayesian inference, maximum entropy, perception, action, decision

A.1 Introduction

The most incomprehensible thing about the world is that it is comprehensible[6]

Albert Einstein

En deçà du langage et de toute activité intellectuelle verbalisable, existe l'immense domaine cognitif de la sensori-motricité. Cet article propose une réflexion à ce sujet.

La démarche suivie est à la fois modélisatrice et expérimentale. Modélisatrice, car nos sujets de réflexions sont des théories mathématiques. Expérimentale, car nous nous attachons à tester et illustrer la pertinence des concepts proposés et des arguments avancés par des expériences concrètes qui soulèvent en retour de nouvelles interrogations. Ces expériences sont rendues possibles, car les modèles mathématiques étudiés peuvent être utilisés pour animer des dispositifs sensori-moteurs artificiels, mécaniques et électroniques, autrement dit des robots.

Jusqu'à quel point agir et percevoir supposent-ils de « comprendre » ou même plus simplement de se « représenter » le monde ? Telle est l'une des préoccupations fondamentales des recherches sur la sensori-motricité. Question élémentaire dont les échos, pourtant, résonnent depuis bien longtemps entre diverses disciplines des sciences cognitives : philosophie, logique, linguistique, éthologie, neurosciences, psychologie ou intelligence artificielle.

Cette question une fois formalisée, expurgée, simplifiée et traduite en termes mathématiques, nous amène à nous interroger tout au long de cet article, sur les liens qui peuvent exister entre les inférences formelles mécanisées informatiquement et leurs contreparties dans le monde physique où évolue un robot. Ainsi reformulée, la question centrale débattue devient : comment rendre effectives les inférences formelles ?

A.1.1 Ceci est une pipe - Ceci n'est pas une pipe

Le paradigme[7] dominant en programmation des robots, comme en science cognitive, est celui de la modélisation de l'environnement. La modélisation d'un phénomène suppose 2 éléments indissociables :

- Un système formel de manipulation de signes qui mécanise le raisonnement
- Une *interprétation*[8] qui définit mathématiquement une analogie entre le système formel et un phénomène modélisé. L'*interprétation* met en relation les signes utilisés avec les objets et ensembles d'objets étudiés. L'*interprétation* est indispensable, car c'est elle qui est censée rendre effectifs les calculs du système formel en permettant de les coupler

---

6 La chose la plus incompréhensible a propos du monde est qu'il soit compréhensible

7 Au sens de Kuhn : noyau dur épistémologique, théorique, formel et méthodologique dont les qualités sont telles qu'il suscite l'adhésion consensuelle et sans restriction d'une large majorité des chercheurs d'un domaine donné à un moment donné.

8 AVERTISSEMENT : Pour une meilleure compréhension du texte, le lecteur devra prendre garde, que le mot "interprétation" sera toujours employé avec ce sens mathématique précis (définition donnée paragraphe II.1) et jamais dans aucun de ses sens communs. En guise d'aide-mémoire, "interprétation" apparaîtra en italique dans tout le texte.

avec l'environnement. Nous verrons cependant, que la notion d'*interprétation* fait nécessairement appel à celle de « domaine de discours » qui suppose d'être capable de « catégoriser » le phénomène, c'est-à-dire d'identifier des objets et ensemble d'objets. Pour les systèmes sensori-moteurs, la « catégorisation » passe nécessairement par le biais de la perception et de l'action et va s'avérer être la difficulté centrale.

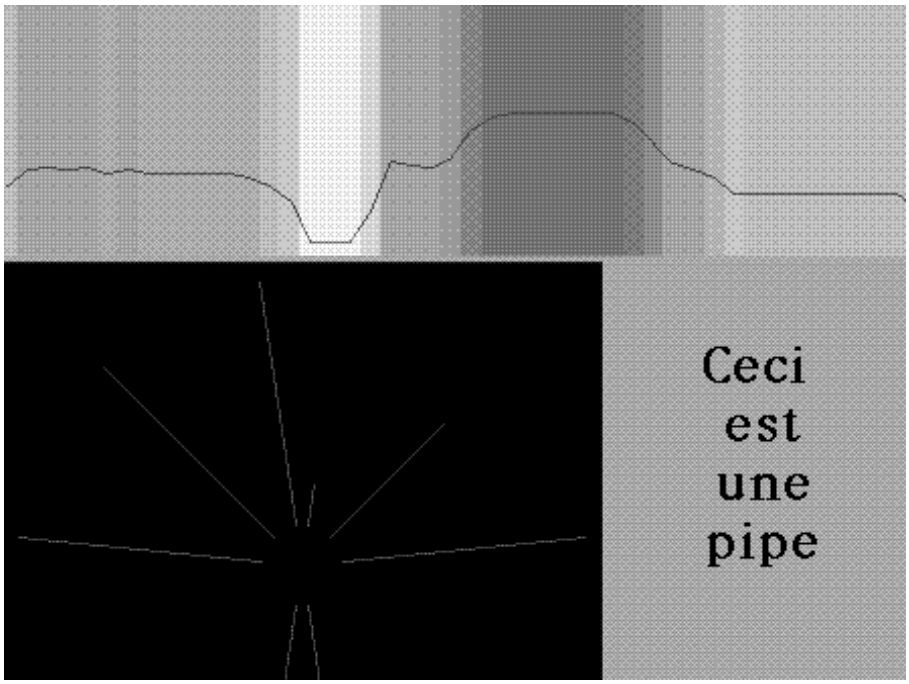


Figure 1 : ceci est une pipe

Figure 2 : ceci n'est pas une pipe

Dans la suite, nous appellerons ce paradigme « l'approche F+I » (système Formel + *Interprétation*).

Une telle approche, du fait de l'existence nécessaire d'un domaine de discours, suppose donc implicitement d'admettre les 2 postulats suivants :

- Le robot est capable, tout comme son concepteur, chacun à travers son appareil sensori-moteur, de percevoir une STRUCTURE ENSEMBLISTE (objets et ensemble d'objets) de l'environnement.
- Le robot et son concepteur peuvent percevoir la MÊME structure ensembliste.

Dans le paragraphe II, nous contestons les fondements de cette approche en mettant en doute ces deux postulats. L'argumentation est double et peut se résumer par la formule, « Ceci est une pipe - Ceci n'est pas une pipe ».

La figure 1 montre comment l'un de nos sujets d'expérience, le robot Khépéra, « voit » une pipe. On conçoit aisément les difficultés que peuvent avoir Khépéra et son concepteur pour se mettre d'accord sur l'ensemble des objets qu'il convient de ranger dans la catégorie « pipe ». Les différences entre les appareils sensori-moteurs du programmeur et du robot rendent en grande partie illusoire qu'ils puissent partager une même structure ensembliste de l'environnement (postulat 2).

La figure 2, célèbre tableau de Magritte, montre la difficulté à répondre à une question en apparence simple : « qu'est ce qu'une pipe ? ». En fait la difficulté de répondre provient de ce que la question même est ambiguë. Elle n'a guère de sens en soi : c'est le contexte qui lui donne un sens. « Ceci » n'est certainement pas une pipe si vous voulez la fumer, mais que répondrez-vous si on vous la montre en vous demandant ce que c'est ? Si une pipe est brisée en deux, pourrez-vous la fumer ? Certainement pas, serait-elle moins « pipe » pour autant ? En combien de morceaux faudrait-il qu'elle soit brisée pour perdre sa qualité de pipe ? Les magnifiques et brûlants narguilés qu'on vous propose dans les ruelles d'Istanbul, sont-ils des pipes ? Sans doute, si vous ressentez une nécessité de fumer. Certainement pas s'il s'agit d'en ranger un dans votre poche. En définitive, il est impossible a un être sensori-moteur de rester constant quant à son propre jugement de ce qu'est une pipe.

L'ensemble des pipes n'est pas clairement défini. « Ceci n'est pas une pipe » va donc nous entraîner très loin. En remettant en cause la validité du premier postulat, nous nous attaquons aux fondements même de la formalisation. En effet, le concept d'appartenance à un ensemble est parfaitement dual de la notion de vérité. Il y a toujours équivalence entre la vérité d'une proposition et la définition d'un certain ensemble. Munir un phénomène d'une structure ensembliste revient à établir la vérité (ou la fausseté) des propositions correspondantes. Finalement, ce que nous remettrons en cause, c'est qu'il soit possible pour des êtres sensori-moteurs de parler et d'accéder à des vérités objectives indépendantes de leurs connaissances préalables et de leurs états sensori-moteur et mental.

A.1.2 Etre pipe ou ne pas être pipe, telle n'est pas la question

F+I est l'aboutissement moderne de la très ancienne association entre raison et discours. Le système formel F est un « langage » avec lequel il est à la fois possible de « parler » et de « penser » l'environnement.

Une alternative à cette conception langagière de la pensée a toujours existé. Elle associe pensée et image. Dans ce cadre, nous proposons une théorie alternative des systèmes cognitifs sensori-moteurs, baptisée approche F+D (système Formel + *Description*). L'essence de cette approche est de proposer de rendre effectifs les calculs formels par un autre moyen que le

couple « *interprétation* - domaine de discours ». Nous avons appelé *description*[9] cet autre moyen.

L'approche F+D est introduite au paragraphe III. La théorie mathématique qui la sous-tend est présentée dans un autre article de ce même numéro intitulé « *Interprétation ou Description (II) Fondements mathématiques de l'approche F+D ».*

L'approche F+D suppose 2 éléments indissociables :
- un système formel ou système de manipulation de signes qui mécanise le raisonnement ;
- une *description* qui définit mathématiquement une relation entre le système formel et le domaine de valeurs des variables sensori-motrices et internes du robot appelé espace des phases. Une *description* permet, ainsi, de rendre effectifs les calculs du système formel en les couplant directement avec les capteurs et les actionneurs du robot. F+D n'a pas recourt à un domaine de discours, il n'est pas nécessaire de projeter de structure ensembliste sur le phénomène observé.

Dans l'approche F+I les signes manipulés par le système formel sont des symboles correspondant à des objets et classes d'objets de l'environnement. Dans l'approche F+D les signes utilisés sont les valeurs des variables sensorielles, motrices et internes du robot. Lorsque le robot interagit de manière pertinente avec son environnement toutes ces variables ne sont pas indépendantes, certaines varient de manières étroitement corrélées. Par exemple, lorsqu'on fume une pipe, les sensations olfactives, tactiles, kinesthésiques et visuelles ne sont pas quelconques. Odeur du tabac, chaleur, poids de la pipe et fumée forment un tout perceptif cohérent. La perception des unes rend probable celle des autres. Que l'une manque ou diffère, révèle en général un fait important. Les signes manipulés par le système formel, c'est-à-dire les valeurs de ces différentes variables, ne sont pas des symboles : ils n'ont, pris isolément, aucune contrepartie dans l'environnement, aucune valeur distale. Aucune des informations transmises par le nez, la main, le bras ou la rétine, n'est en soit significative. Pas plus que la couleur d'un pixel sur une rétine n'a de signification et ne permet de se faire une idée quelconque de l'image. C'est l'ensemble des interdépendances entre ces variables qui est caractéristique et que nous avons baptisé « *description* ». La connaissance des valeurs de certaines d'entre elles permet de prédire les valeurs probables des autres. Lorsqu'on saisit une pipe qui fume, on s'attend à la sensation de chaleur. Ceci s'applique aussi bien aux variables motrices qu'aux variables sensorielles. Si l'on tient une pipe allumée, il est probable que les prochains ordres moteurs envoyés au bras soient pour la porter à la bouche. Être pipe ou ne pas être pipe, telle n'est plus la question. Seule reste la question de savoir quelles actions peuvent être prises étant donné la situation considérée.

Dans l'approche F+I, l'*interprétation* et le domaine de discours sont donnés a priori par le concepteur. Dans l'approche F+D les *descriptions* sont apprises, certes sur la base de connaissances préalables fournies par le concepteur, mais surtout petit à petit, par la somme des expériences cumulées du robot. Construire une *description* va consister pour le robot à élaborer des structures internes rendant compte des dépendances observables. Utiliser l'approche F+D lui permettra de percevoir, décider et agir en menant des calculs formels basés sur ces structures internes acquises expérimentalement.

L'approche F+D repose sur un fondement mathématique parfaitement clair : les probabilités. Nous proposons de coder les dépendances entre variables sensorielles, motrices et internes, i.e. ce que nous appelons : *descriptions*, par des distributions de probabilités.

---

9 AVERTISSEMENT : Pour une meilleure compréhension du texte, le lecteur devra prendre garde, que le mot "description" sera toujours employé avec ce sens mathématique précis (définition donnée paragraphe III.2) et jamais dans aucun de ses sens communs. En guise d'aide-mémoire, "description" apparaîtra en italique dans tout le texte.

Nous montrons comment les règles de l'inférence probabiliste permettent de faire des raisonnements aussi riches et complexes que ceux faits avec les règles formelles logiques fondant l'approche F+I qui apparaît ainsi comme une approximation de F+D. Nous explicitons comment ces calculs formels probabilistes sont rendus effectifs par la prédiction des valeurs probables des variables motrices. Nous expliquons comment le principe de maximum d'entropie fournit un cadre théorique général et clair pour gérer et comprendre les problèmes d'apprentissage.

A.2 Critique de l'approche F+I des systèmes sensori-moteurs

> *Mais le rapport du langage à la peinture est un rapport infini. Non pas que la parole soit imparfaite, et en face du visible dans un déficit qu'elle s'efforcerait en vain de rattraper. Ils sont irréductibles l'un à l'autre : on a beau dire ce qu'on voit, ce qu'on voit ne loge jamais dans ce qu'on dit, et on a beau faire voir, par des images, des métaphores, des comparaisons, ce qu'on est en train de dire, le lieu où elles resplendissent n'est pas celui que déploient les yeux, mais celui que définissent les successions de la syntaxe.*
>
> *Michel Foucault*[10]
>
> *... les terrifiants pépins de la réalité !*
>
> *Jacques Prévert*

A.2.1 Principe de l'approche F+I

Concevoir la théorie formelle d'un certain phénomène consiste à produire un système de manipulation de signes obéissant à des règles purement syntaxiques et dont le comportement présente des analogies intéressantes avec le phénomène étudié. Un tel système de manipulation de signes est appelé système formel.

L'analogie entre le système de manipulation de signes et le phénomène est donnée par une *interprétation* qui met en relation les signes utilisés et les objets et ensemble d'objets étudiés.

La caractéristique la plus fondamentale et la plus intéressante des systèmes formels est qu'ils peuvent exister et fonctionner par eux-mêmes, indépendamment du phénomène qu'ils sont censés représenter. On peut produire de manière systématique, à partir de suites de signes et à partir de règles syntaxiques, de nouvelles suites de signes. Point besoin pour cela d'attacher aucune espèce de signification à aucun des signes. Point besoin de savoir de quoi on parle pour pouvoir en parler avec certitude ! Cette caractéristique permet d'assurer qu'en donnant le même système formel à deux mathématiciens (ou à deux machines), ils (elles) seront capables de produire les mêmes suites de signes. Voilà qui assure ce qu'on appelle communément la rigueur d'une théorie. Voilà mise en pratique et en équation l'ambition aristotélicienne et leibnizienne de voir inéluctablement se mettre d'accord les différents « raisonneurs » partant des mêmes prémisses.

On peut envisager de produire et d'étudier des systèmes purement formels en dehors de toute analogie avec une quelconque autre réalité. Cependant, l'intérêt de ce « jeu » syntaxique

---

10 « *Les mots et les choses* » [Foucault66]

n'est pleinement révélé que par la donnée d'une *interprétation*, quand les objets et ensemble d'objets étudiés peuvent être dénotés par des signes, les règles de manipulation de signes mises en correspondance avec des opérations sur ces objets et ces classes, et les chaînes de signes produites interprétées en terme de prédictions (ou d'explications) sur le comportement du phénomène étudié. Alors, et alors seulement, nous aurons une théorie formelle de notre phénomène.

L'*interprétation* est une notion mathématique bien définie. Elle peut même être, elle-même, formalisée comme le prouvent les travaux de Tarski et Carnap sur la sémantique extensionnelle et la théorie des modèles. Une *interprétation* est définie par :

- l'ensemble des objets du phénomène modélisé appelé domaine de discours Δ ;
- une application (X) de C (l'ensemble des symboles de constante du système formel) dans Δ, qui associe à chaque symbole de constante un objet de Δ ;
- une application (Π) de P (ensemble des symboles de prédicat du système formel) dans l'ensemble des relations sur Δ (classes d'objets) qui associe à chaque symbole de prédicat n-aire une relation n-aire sur Δ.

La notion d'*interprétation* suppose donc l'existence d'un domaine de discours. L'existence de ce domaine de discours postule implicitement que le phénomène modélisé ait une structure ensembliste et que cette structure soit « accessible » aux « utilisateurs » du système formel. Il faut, en effet, que ces utilisateurs puissent définir rigoureusement (mathématiquement) les objets de Δ, les ensembles d'objets de Δ et les ensembles d'ensembles d'objet de Δ (une relation n-aire est un ensemble d'ensembles à n éléments d'objets de Δ). De façon duale[11], il faut que ces utilisateurs puissent définir des valeurs de vérité pour toute relation entre éléments de Δ correspondant à un prédicat défini dans le système formel. S'ils ont cette capacité, alors, les utilisateurs du système formel pourront se servir de sa mécanique syntaxique pour démontrer des « théorèmes » qui seront, sans doute, pour eux interprétables comme des « vérités » dans le domaine de discours et pour le phénomène modélisé.

Nous appellerons « catégorisation » l'opération consistant, pour chaque utilisateur du système formel, à « projeter » sur le phénomène cette structure ensembliste (voir figure 3).

Si l' *interprétation* est une notion mathématique parfaitement définie, la catégorisation, elle, est une notion cognitive qui prend en compte nécessairement les capacités perceptives, intellectuelles et motrices de l'utilisateur considéré. Adopter cette définition de l'*interprétation*, et notamment le concept de domaine de discours, c'est admettre que cette catégorisation soit possible. Non seulement qu'elle soit possible, mais qu'elle puisse être mise en œuvre avec toute la rigueur nécessaire, afin de définir mathématiquement les limites des ensembles correspondant aux symboles de constantes et de prédicats du système formel.

Ce que nous apprend le domaine de la perception et de l'action, c'est que cette thèse est trop restrictive lorsqu'il s'agit d'interagir avec un monde non intrinsèquement formel, comme c'est le cas pour les robots. Ce que nous remettons en cause au paragraphe II.4, c'est la nécessité même de catégoriser l'environnement en termes d'un domaine de discours prédéfini. Ni les logiques multi-valuées ou plausibilistes, ni même la modélisation probabiliste, ne résout rien tant que subsiste cet intermédiaire obligé d'un domaine de discours distal. C'est aussi ce que nous enseigne les si anciennes, si nombreuses et si subtiles réflexions épistémologiques sur la triade « réalité - observateur - représentation » et son pendant « référent - signifié - signifiant ».

---

[11] Cette dualité est à mettre en parallèle avec celle entre « Théorie des modèles » et « Théorie de la démonstration » [Kleene67].

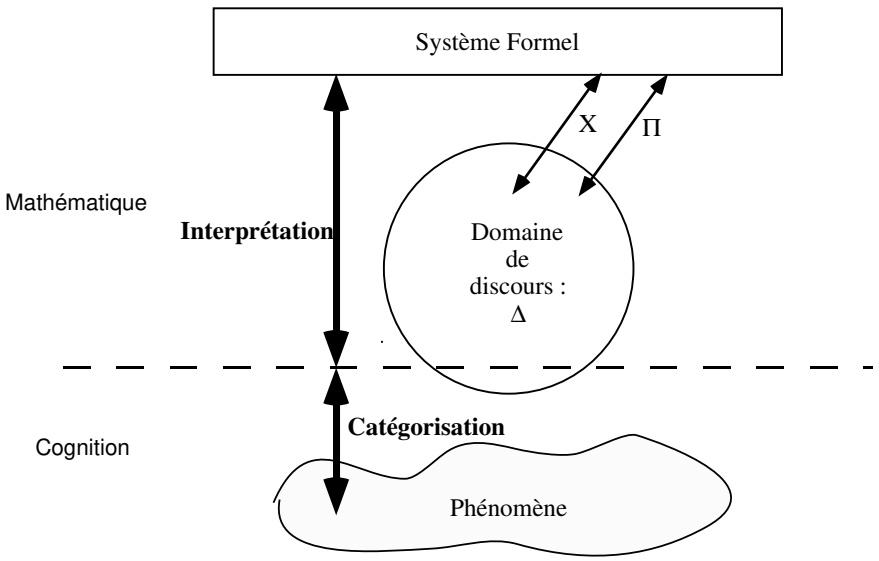


Figure 3 : principe de l'approche F+I

A.2.2 Les systèmes formels comme langages des sciences

Au cours du XIX^e^ et du XX^e^ siècle, les théories formalisées prennent un essor considérable. Toutes se formalisent, la mathématique bien sûr, la physique, la chimie, l'astronomie, toutes les sciences dites « dures », mais aussi des pans entiers des sciences de la vie ou des sciences de l'homme. La linguistique, par exemple, voit se développer une énorme branche entièrement formalisée, l'économie ou la sociologie aussi. Même si pour de nombreuses sciences une formalisation précise ou exhaustive n'est pas possible, l'influence du paradigme formel est partout et les méthodes de raisonnement correspondantes deviennent un passage quasi obligé à la plupart des démarches scientifiques. Le succès est à la clef et l'extraordinaire progrès de la science moderne est en grande partie dû à la démarche formalisatrice.

Le succès de cette démarche s'explique aisément, ses avantages sont en effet multiples :

- garantir une rigueur « maximum » ;
- permettre une certaine « mécanisation » du raisonnement qui écarte une grande partie des sources d'erreur ;
- inciter à la réduction, l'abstraction et la simplification ;
- offrir un langage commun, supposé non ambigu, pour communiquer et débattre ;
- obliger à un jugement intrinsèque sur la validité de la théorie avancée, par exemple, en examinant des critères tels que consistance et complétude;
- codifier le jugement extrinsèque sur la qualité des prédictions fournies ;
- enfin, et peut-être surtout, conduire à de formidables synthèses du savoir et d'éblouissants progrès scientifiques lorsque le même système formel s'avère avoir plusieurs significations différentes, c'est-à-dire lorsque deux phénomènes parfois en apparence complètement différents ont en commun certaines structures ou certains principes de fonctionnement.

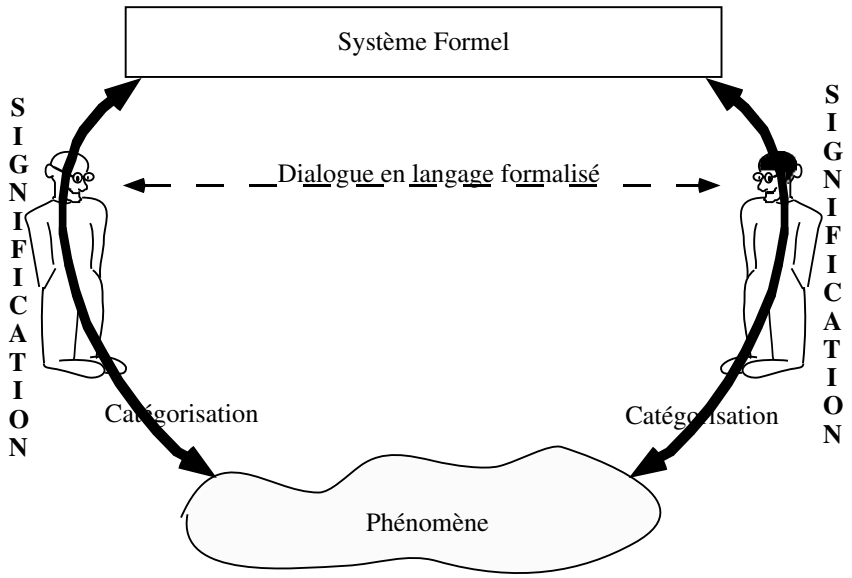


Figure 4 : les systèmes formels comme langage des sciences

Cependant, si l'on regarde de près ces diverses tentatives de formalisation, on s'aperçoit que dans la très grande majorité des cas, seule la première moitié du chemin a été parcourue. Un système formel est construit, mais son *interprétation* et son domaine de discours ne sont pas donnés de manière explicite et rigoureuse. La catégorisation est laissée à l'appréciation de chacun. Qu'à cela ne tienne ! Le système formel est dans ce cadre utilisé par les scientifiques pour communiquer et argumenter. Ils se comprennent mutuellement si, et seulement si, ils attachent des significations assez proches les unes des autres à ce système. De ce point de vue, ils sont alors dans une situation de communication tout à fait comparable à celle que l'on rencontre avec l'usage du langage naturel. D'ailleurs, comme pour le langage naturel, les situations où la communication est intéressante sont celles ou les compréhensions de chacun sont assez proches pour permettre l'échange, tout en étant assez différentes pour susciter le débat. Si l'*interprétation* était spécifiée rigoureusement, le débat scientifique n'aurait pas de raison d'être. Deux scientifiques ne peuvent qu'être d'accord sur un raisonnement purement formel. Leur seule latitude pour débattre est dans la signification que chacun d'eux attache aux résultats d'un raisonnement formel. La signification de ces résultats et calculs (théorèmes, équations, résultats numériques) est laissée à l'appréciation de chacun, qui dispose de son sens commun, de sa culture du domaine, de son appareillage expérimental et de ses sens pour juger de leur pertinence et de leur porté. C'est là que s'exprime réellement la compétence de ces hommes, par leurs capacités intimes à comprendre, appréhender et traduire les résultats formels en termes de prédiction sur le phénomène modélisé.

A.2.3 L'informatique : une nouvelle dimension de la formalisation

L'importance de la démarche F+I a acquis une nouvelle dimension au cours de la seconde guerre mondiale avec la naissance de l'ordinateur. D'une part, parce que l'informatique est l'incarnation électronique du concept de système formel. Données et programmes, sont les éléments constitutifs d'un système formel, et l'ordinateur, la « mécanique » qui automatise les inférences. D'autre part, parce que l'informatique a ouvert des possibilités entièrement nouvelles à la démarche F+I. Avoir une théorie scientifique formalisée permettant d'élaborer de nouveaux modèles et de concevoir des dispositifs technologiques est une chose, mais disposer de modèles formels, interrogeables en temps réel, pour prendre des décisions ou commander une machine en est une tout autre !

Les programmes d'un ordinateur peuvent être vus comme un ensemble de systèmes formels communiquant les uns avec les autres. Quand il s'agit de faire communiquer entre eux deux programmes, spécifier une *interprétation* est une tâche relativement facile, étant donné que ces deux programmes sont de nature formelle et donc, par essence, catégorisable.

Reste le problème du lien entre les différents systèmes formels (programmes) présents dans l'ordinateur et le phénomène modélisé. Là encore, on constate que cette *interprétation* n'est pas donnée de manière explicite et rigoureuse. La machine communique avec son utilisateur en langage formalisé, sans avoir aucunement besoin de rattacher ses inférences au phénomène (voir figure 5). Ce dialogue formalisé repose sur le système formel correspondant à l'interface homme-machine. Cette communication est possible, car l'utilisateur et l'ordinateur partagent une même *interprétation* de ce système formel. Ils sont capables d'identifier les mêmes objet et ensembles d'objets dans l'environnement formel de ce dialogue. Par exemple, l'utilisateur d'un traitement de texte et le programme correspondant savent reconnaître les mêmes caractères, ils sont d'accord sur ce qu'est un mot (une suite de caractères entre deux blancs ou entre un blanc et un signe de ponctuation) ou sur ce qu'est un alinéa. Tout l'intérêt de l'informatique vient de ce que l'utilisateur accorde aux objets et ensemble d'objets manipulés

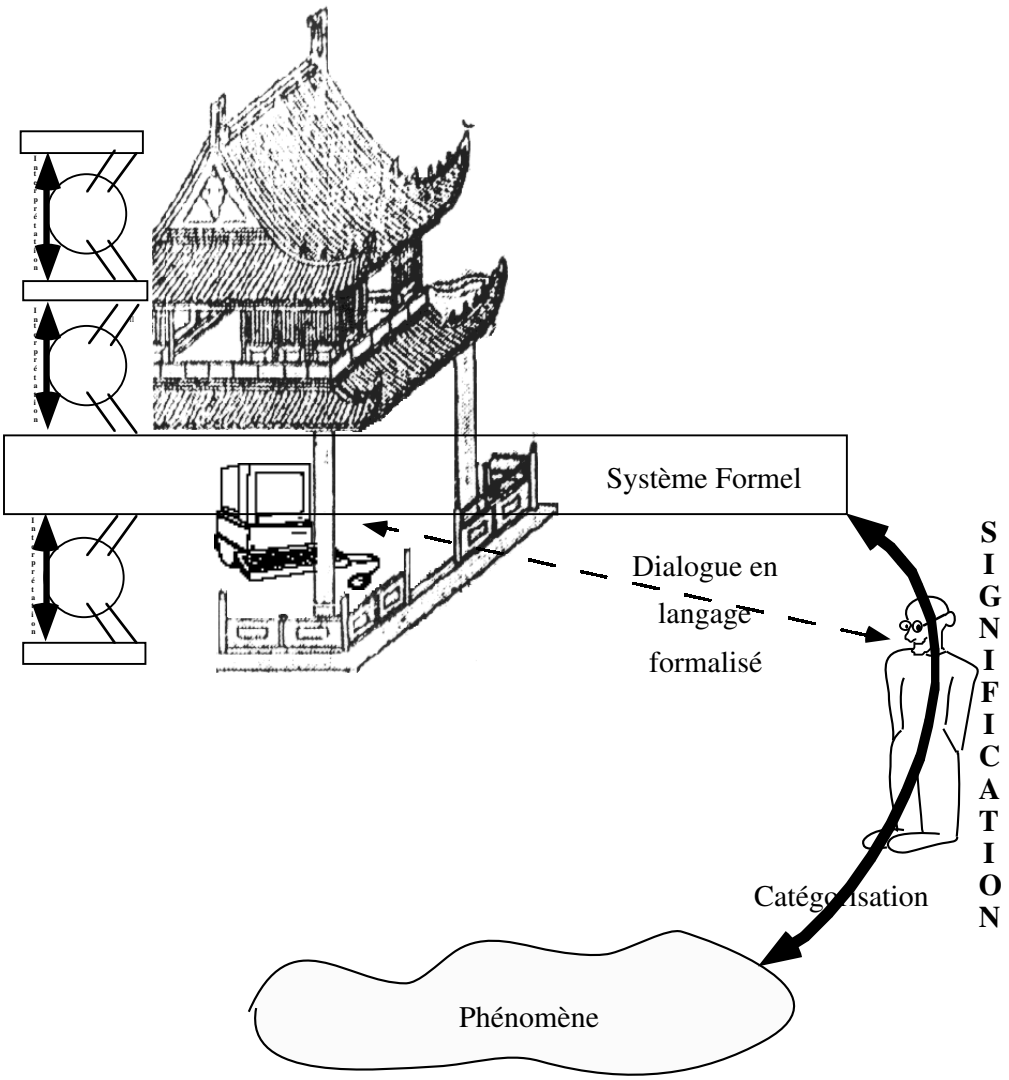


Figure 5 : l'ordinateur dans sa « chambre chinoise »

Il y manipule les « idéogrammes » mais pas les « idées » correspondantes.

au cours de ce dialogue une autre signification. Il attribue aux manipulations formelles effectuées par la machine, une signification relative au phénomène modélisé. Cette signification échappe totalement aux programmes - alors que c'est celle qui importe à l'utilisateur. C'est le sens du texte et l'esthétique de la mise en page qui compte pour l'utilisateur du traitement de texte, mais il est bien content d'avoir l'ordinateur pour l'aider à en manipuler la forme.

A.2.4 Difficultés de l'approche F+I des systèmes sensori-moteur

Les deux paragraphes précédents ont montré comment l'approche F+I échappait à la nécessité de spécifier une *interprétation* lorsqu'on l'applique, soit aux sciences, soit à l'informatique. Quand l'approche F+I est appliquée à un système sensori-moteur, comme nous allons le voir, il n'est plus possible d'échapper à la spécification rigoureuse d'une *interprétation*. Cette approche se heurte donc dans ce cas, de plein fouet, aux insurmontables difficultés de la catégorisation.

Le phénomène modélisé par un système sensori-moteur est son propre environnement physique. Le système interagit directement avec cet environnement, il fait partie intégrante du phénomène qu'il modélise. Cette interaction ne passe plus par la médiation d'un utilisateur humain, mais par le biais des capteurs et actionneurs du robot. Dès lors, pour que les calculs formels effectués par le robot puissent se traduire en termes sensori-moteurs, il faut que chaque signe manipulé réfère à un objet ou une classe d'objets de l'environnement. L'*interprétation* du système formel devient indispensable. Il doit donc exister un programme capable de faire la catégorisation. C'est-à-dire capable de combler l'immense fossé séparant, d'une part, les signaux bruts arrivant des capteurs et envoyés aux actionneurs et, d'autre part, la *description* abstraite et rigoureuse de l'environnement que le système formel suppose.

Le programmeur a sa propre conception de l'environnement. Via ses sens, ses instruments et vu l'ensemble des expériences et des connaissances qu'il a accumulées au cours de sa vie, il est capable de « projeter » une structure ensembliste sur l'environnement. Il peut nommer les objets et ensemble d'objet qu'il observe. Il peut construire un modèle formel de l'environnement et attacher, à chacun des termes et à chacune des propositions de ce modèle, une signification (i.e. les relier à l'environnement via ses perceptions et ses actions). Il est capable pour certaines des propositions du système formel de dire (ou de faire calculer par une machine) si elles sont dérivables ou réfutables. Il est en mesure de déterminer si la signification de ces formules correspond à une vérité ou une contrevérité dans l'environnement.

Le programmeur utilise ce système formel pour programmer le robot, c'est-à-dire pour lui spécifier ce qu'il doit faire et comment il doit le faire. Il l'utilise aussi pour recueillir les informations que le robot lui renvoie. Il utilise donc le système formel comme un langage pour communiquer avec le robot. Toutefois, ils ne peuvent tous deux se « comprendre » que si la signification, que le programmeur attache aux signes et aux propositions du système formel, est assez proche de l'*interprétation* de ces signes qu'il a fournie au robot. Concrètement, cela signifie que le programmeur a dû fournir au robot un programme lui donnant la faculté de catégoriser correctement l'environnement à partir de ses données sensori-motrices. « Catégoriser correctement » signifie que le robot est capable, à partir de ses données sensori-motrices propres, d'identifier, sans aucune ambiguïté ni incertitude, les objets et ensembles d'objets du domaine de discours sous-tendant le langage formel utilisé par le programmeur pour dialoguer avec lui (voir figure 6).

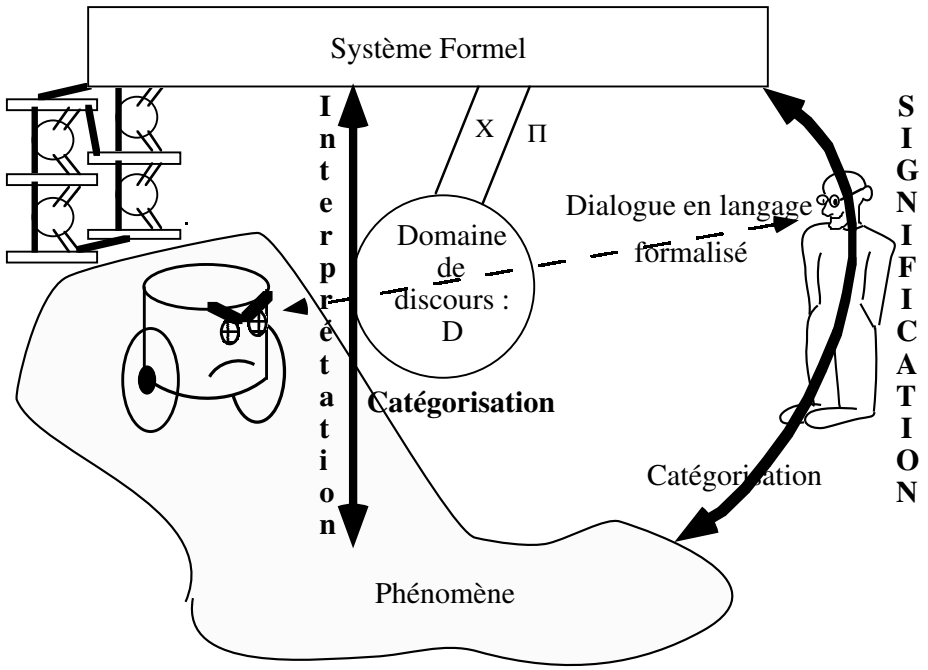


Figure 6 : difficultés de l'approche F+I des systèmes sensori-moteur

Une première difficulté dans l'écriture de ce programme tient aux différences existant entre les appareils sensori-moteurs du robot et du concepteur. Le concepteur voudrait que le robot soit capable de distinguer les mêmes objets et ensembles que lui. Cela peut-il être possible alors qu'ils ont des capteurs et des actionneurs aussi différents ?

La principale des difficultés n'est cependant pas là. Un système formel lorsqu'il sert à décrire un environnement physique ne peut pas être complet. Il n'est pas possible de prendre en compte exhaustivement les facteurs influençant le phénomène étudié, pour au moins trois raisons de principe :

- l'idée même d'exhaustivité est en contradiction avec le concept de modèle. Un modèle n'est intéressant que s'il est (beaucoup) plus simple que le phénomène étudié.
- les lois de la physique font que très rapidement il faudrait tenir compte de l'univers entier pour pouvoir décrire un phénomène physique même en apparence simple.
- les systèmes dits « chaotiques » prouvent qu'il est impossible de fournir de modèle « simplifié » de certains phénomènes, même élémentaires, dans la mesure où il faut connaître « exactement » leurs conditions initiales pour être capable de prédire leur évolution.

En pratique, il n'est pas besoin d'invoquer ces raisons fondamentales. Un système sensori-moteur, qu'il soit vivant ou artificiel, doit de manière évidente prendre des décisions en n'ayant qu'une connaissance extrêmement partielle de son environnement.[12]

Dès lors, il est impossible de catégoriser de manière absolue à partir des données sensorielles et motrices. Le modèle utilisé pour interpréter les données n'est jamais suffisamment précis et complet pour pouvoir rendre compte de toutes les variations possibles des mesures. Il existe toujours des variables « cachées » qui « brouillent » les données sensori-motrices. Le diagnostic souvent porté sur ce genre de situation consiste à dire que les

12 Les systèmes vivants même les plus simples nous prouvent que cela est possible.

données sensori-motrices sont « bruitées », voire même quelquefois « aberrantes ». Étrange retournement de causalité qui semble considérer le modèle comme exact et imputer au monde physique quelque tare inconnue.

Une des premières expériences que nous avons menées permet d'illustrer de manière très simple ce problème fondamental.

Le dispositif expérimental consiste en un « robot » qui se limite à un axe de rotation vertical commandé en position (angle θ) portant une cellule photoélectrique mesurant une intensité lumineuse (intensité I). L'environnement est un seau de plastique vert dans la paroi duquel on a fixé une lampe éclairant l'intérieur (figure 7).

Nous avons « exploré » l'environnement du seau avec le protocole consistant à tirer au hasard (suivant une loi uniforme) des angles θ, à faire tourner l'axe en fonction de cette commande et à mesurer l'intensité ι lue par la cellule après déplacement. Nous avons ainsi obtenu l'histogramme des couples de valeurs (θ,ι) observés présenté figure 8. La signification de cet histogramme est très claire. Nous observons de hautes valeurs de l'intensité et une saturation de la cellule photoélectrique pour les angles où la cellule fait face à la lampe. Nous observons un deuxième pic d'intensité à 180° du précédent correspondant à la réflexion principale de la lumière sur les parois du seau.

Il est possible de construire de nombreux modèles formels de cet environnement, depuis de simples modèles géométriques jusqu'à de complexes modèles optiques cherchant à décrire précisément les réflexions à l'aide, par exemple, de lancer de rayons.

Sur le papier, le problème possède une symétrie évidente autour de l'axe principal d'illumination de la lampe. En effet, que le robot tourne d'un angle θ vers la gauche ou vers la droite à partir de cette position de référence devrait conduire à des situations parfaitement identiques en ce qui concerne l'illumination de la cellule. Un modélisateur ne manquera certainement pas de tenir compte d'une telle symétrie. Le modèle résultant prédira donc des valeurs identiques pour ι dans ces 2 situations. Or, nous constatons que cette symétrie n'est pas vérifiée par les données expérimentales.

La première idée qui vient à un ingénieur dans ce genre de situation est de penser à un problème de calibration. Un modèle est dit « calibré » quand les valeurs des paramètres du modèle permettent une *description* « satisfaisante » du phénomène. Ici, par exemple, on peut penser que les paramètres géométriques ne sont pas exactement les bons, que l'axe vertical n'est peut-être pas parfaitement vertical ou peut être pas parfaitement centré. On peut alors, soit aller réaligner l'axe vertical, soit complexifier le modèle pour tenir compte du centrage et de l'inclinaison de l'axe. Nous avons essayé de le faire sans y parvenir. Les raisons de l'absence de symétrie sont ailleurs. Quelles sont-elles exactement ? Nous n'en savons rien. De nombreuses possibilités sont imaginables :

- la cellule présente des phénomènes d'hystérésis qui font que les valeurs lues dépendent de l'illumination mais aussi du « passé » des mesures effectuées précédemment (cette

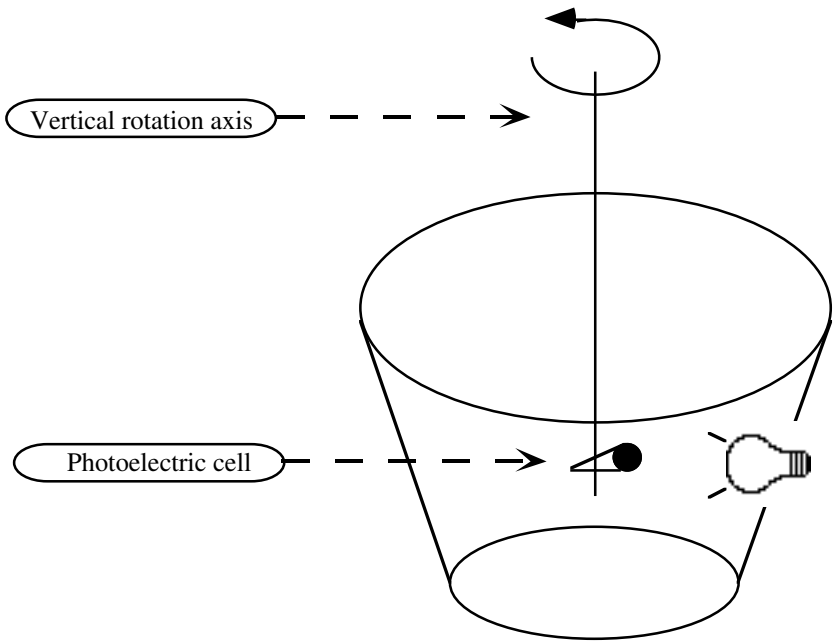


Figure 7 : dispositif expérimental

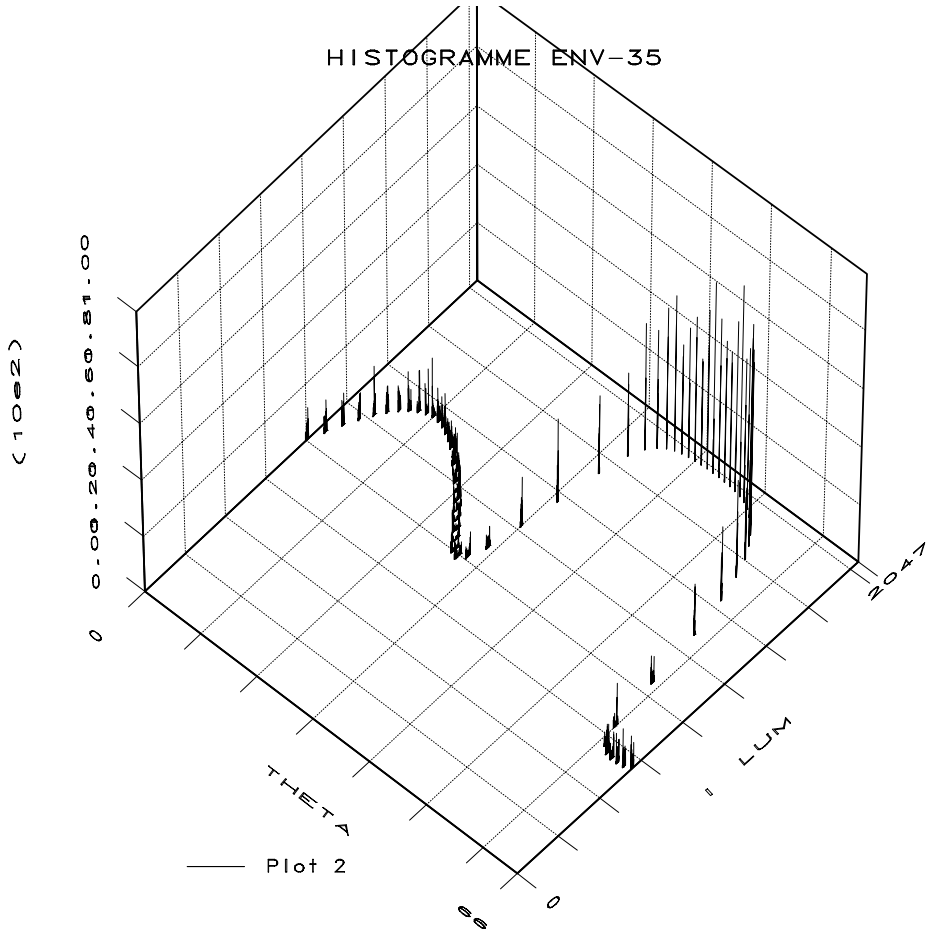


Figure 8 : histogramme des couples de valeurs (θ,ι) observés

explication peut paraître compliquée, elle s'est pourtant avérée partiellement vraie) ;
- un néon était allumé dans la salle au moment de l'expérience, brisant la supposée symétrie de l'illumination ;
- la commande angulaire du robot est plus imprécise pour certains angles que pour d'autres, les angles réellement atteints n'étant plus alors exactement les θ commandés (cette explication semble, elle aussi, bien compliquée, et elle s'est, elle aussi, avérée partiellement vraie, un défaut électronique bloquant à 0 certains bits) ;
- le seau est plus poussiéreux à certains endroits qu'à d'autres ;
- etc..

Complexifier le modèle pour pouvoir tenir compte de ces différentes causes possibles devient, manifestement, beaucoup plus difficile. Il faudrait modéliser l'électronique de la cellule, celle du robot, l'éclairage de la salle et le comportement de la femme de ménage. Finalement, on voit clairement par cette expérience que, même pour un système sensori-moteur et un environnement aussi simple, il est très délicat de prétendre fournir un modèle formel exhaustif et complet de leur interaction.

Cette difficulté à faire le lien entre les signes du système formel et les données sensori-motrices est largement reconnue en intelligence artificielle et en robotique, où elle prend de multiples visages et divers noms, dont nous ne retiendrons que celui de « problème de l'ancrage perceptif des symboles »[13].

La parade habituelle pour pallier cette difficulté consiste à contraindre l'environnement dans lequel évolue le système sensori-moteur et les fonctions qu'il doit remplir. Dans l'exemple précédent, comme on l'a déjà dit, le réflexe d'un roboticien serait de rechercher les causes de la dissymétrie pour modifier soit le robot, soit l'environnement, afin de supprimer ce « défaut ». « Défaut » signifiant ici que le monde physique a le mauvais goût de ne pas vouloir correspondre au modèle mathématique supposé le décrire. En robotique industrielle, cette approche est légitime et permet de développer des applications robotiques opérationnelles. L'environnement du robot peut toujours être étroitement contrôlé et les tâches qu'il doit remplir bien spécifiées. C'est cela qui assure la pertinence du modèle employé.

Contrôler et contraindre l'environnement n'est cependant pas toujours possible. Les systèmes sensori-moteurs vivants n'évoluent pas dans des environnements spécifiquement conçus pour eux. Étant donné les buts scientifiques de ce travail, nous nous intéressons à la robotique en « environnement naturel », c'est-à-dire dans des environnements donnés au robot indépendamment de sa propre conception. L'approche F+I des systèmes sensori-moteurs nous semble dans ce cas dans une impasse, car nous ne voyons pas comment surmonter la difficulté à catégoriser. Seule une approche radicalement différente des systèmes sensori-moteurs peut espérer résoudre un tel problème.

A.2.5 Notes bibliographiques

Nous n'avons pas pu dans cet article rappeler la définition « formelle » d'une formalisation. Des exemples introductifs de formalisation peuvent être trouvés dans (Hofstadter79, notamment aux chapitres 1, 2, 3, 7 et 8) ainsi que dans (Nilsson80, aux chapitres 1 et 4). Un ouvrage de référence, non mathématique, sur l'approche F+I en science, d'une richesse et d'une profondeur sans pareille, est certainement (Piaget67a). On y trouve notamment une très belle introduction à la formalisation dans le chapitre de Grize (Grize67). Un ouvrage de

---

[13] symbol grounding problem, voir Harnad (Harnad90)

référence, cette fois-ci mathématique, complet et rigoureux, est sans aucun doute (Kleene67) où on trouve en particulier une intéressante présentation de la théorie du modèle et de la théorie de la démonstration qui clarifie la notion d'*interprétation*. Une présentation synthétique et claire des systèmes formels du point de vue langage est faite dans (Vauquois76).

Nous n'avons pas pu non plus traiter ici le très vaste et très important sujet des difficultés mathématiques rencontrées par la formalisation et les nombreux théorèmes de limitation qui les explicitent. Une introduction amusante sur ce sujet peut être trouvée dans (Hofstadter79). Dans (Piaget67a), l'article de Ladrière (Ladrière67) en fait une bonne analyse. Les démonstrations de la plupart de ces théorèmes peuvent être trouvées dans (Kleene67). Enfin, une discussion des conséquences philosophiques et surtout cognitives de ces théorèmes peut être trouvée dans (Lucas61).

Les difficultés de l'intelligence artificielle et de la robotique avec la notion d'*interprétation* ont fait couler beaucoup d'encre. Une analyse de la logique en tant que langage est présentée dans les deux articles de Léo Apostel (Apostel67a) et (Apostel67b) toujours dans (Piaget67a). On pourra se référer notamment à (Harnad89) et (Harnad90) concernant le « symbol grounding problem ».

Diverses critiques de l'intelligence artificielle sont fondées sur des arguments similaires à ceux proposés ici, par exemple celles de Dreyfus (Dreyfus79), de Searle (Searle81), de Reeke et Edelman (Reeke88), de Malcom et Smithers (Malcom89), de Winograd et Flores (Winograd86), ou enfin, dans l'ouvrage très complet de Bickhard et Terveen (Bickhard95). Pour tenter de surmonter les difficultés que les approches « classiques » (F+I) rencontrent en robotique, Brooks a proposé l'approche réactive (voir (Brooks86a), (Brooks86b), (Brooks89) et (Brooks90)) qui a été à l'origine de nombreux travaux. L'apprentissage et la modélisation, qui au début n'en étaient pas les préoccupations majeures, ont été abordées depuis sous diverses formes (voir, entre autres, (Mataric91), (Koza91), (Brooks91a) et (Horswill93)).

Cette critique de l'approche F+I mériterait d'être précisément située, dans le cadre de l'immense réflexion épistémologique déjà citée, sur la triade « réalité - observateur - représentation » et son pendant « référé - signifié - signifiant ». Contentons-nous de donner un point de départ intéressant avec le livre de Michel Foucault intitulé « *Les mots et les choses* » (Foucault66) qui a le mérite de présenter toute l'évolution historique des concepts de signe et de langage, et avec le livre de Joëlle Proust « *Question de forme* » (Proust86) plus centré sur les temps modernes et beaucoup plus « technique ».

Au cœur du débat actuel sur le sujet se trouve, d'après nous, la notion d'objectivité dans sa double acception : (1) possibilité de juger de la validité d'une théorie indépendamment de toute connaissance sur celui qui l'a formulée, en faisant complètement abstraction de l'observateur, et par conséquent, (2) possibilité qu'à un niveau de description donné on puisse confondre la réalité avec la connaissance qu'on en a. Admettre qu'il puisse exister des modèles indépendants de tout observateur, c'est supposer que le problème de la catégorisation puisse être parfaitement résolu et que l'environnement puisse être muni d'une structure ensembliste indépendamment de toutes considérations sensori-motrices. Toute cette première partie n'a pas eu d'autres buts que de contester cette possibilité.

Le passionnant livre de Roland Omnès (Omnès94) est pratiquement totalement consacré à l'analyse de l'approche formalisatrice en physique et notamment aux difficultés rencontrées. On trouvera aussi de très intéressantes considérations sur ce sujet dans (Poincaré02) même si, évidemment, les progrès de la physique du XX^{e} siècle ne sont pas pris en compte. Ce sujet est aussi amplement abordé dans (Piaget67a) notamment dans les articles d'André Lichnerowicz (Lichnerowicz67), Jean Piaget (Piaget67b), (Piaget67c) & (Piaget67d), Jean Ullmo (Ullmo67)

et Louis de Broglie (Broglie67). Le livre d'Ivar Ekeland (Ekeland84) fait un tour d'horizon passionnant et abordable des difficultés et critiques de l'objectivisme (et du déterminisme) et est agréablement complété par son livre de 91 (Ekeland91). Une introduction intéressante aux sciences cognitives, revenant sur le rôle épistémologique de l'objectivisme, est donné dans (Stewart92).

Enfin, une présentation différente de cette problématique peut être trouvée dans (Dedieu95) (notamment dans le chapitre 2).

A.3 Introduction à l'approche F+D des systèmes sensori-moteurs

> *Far better an approximate answer to the RIGHT question,which is often vague, than an exact answer to the WRONG question,which can always be made precise.*[14]
>
> *John W. Tukey*[15]

Cette citation de John W. Tukey nous résume bien les difficultés rencontrées par l'approche F+I et indique quelle voie suivre pour les surmonter. Comme on l'a vu, l'approche F+I appliquée aux systèmes sensori-moteurs permet de poser des questions précises aux modèles mathématiques décrivant l'environnement des robots et d'obtenir des réponses exactes à ces questions en faisant des calculs formels. Malheureusement, ces modèles mathématiques ont souvent bien peu à voir avec la réalité sensori-motrice d'un robot. Les questions sont rendues précises par la modélisation, mais perdent ainsi, trop souvent, toute pertinence pratique. Il nous faut donc reconsidérer la notion d'*interprétation* en imaginant un autre moyen de rendre effectifs les résultats des calculs. Comme nous allons le voir, le prix à payer pour obtenir ce résultat sera, pour l'approche F+D (Système Formel + *Description*), de ne pouvoir fournir que des réponses approximatives aux problèmes sensori-moteurs. Cependant, mêmes approximatives, ces réponses seront suffisantes pour permettre à un robot d'interagir avec son environnement.

A.3.1 Espace sensori-moteur et espace des phases

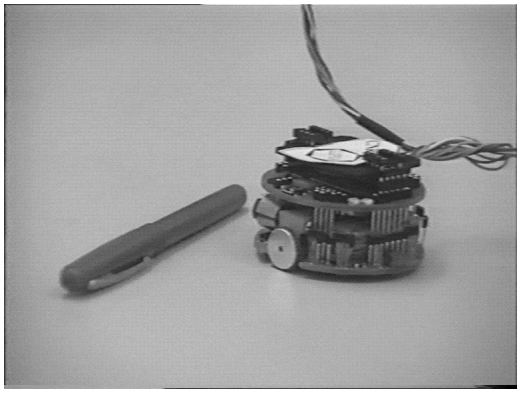
Pour un robot, nous appelons « espace sensori-moteur » l'ensemble des valeurs possibles pour les signaux venant de ses capteurs et envoyés à ses actionneurs.

Le robot peut, en partie, contrôler son interaction avec l'environnement en choisissant des valeurs pour ses variables motrices. L'ensemble des calculs qui le conduit à faire ces choix se base non seulement sur les variables sensori-motrices, mais aussi sur des variables internes, dites variables d'état. Nous appellerons « espace des phases », l'ensemble des variables manipulées par le système au cours de ses raisonnements. L'espace sensori-moteur est un sous-ensemble de l'espace des phases. L'espace des phases est hautement structuré. Quand les valeurs de certaines des variables sont imposées, les valeurs possibles pour certaines autres sont fortement contraintes.

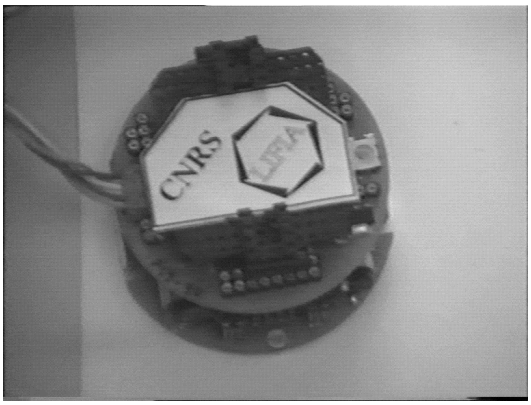
---

[14] Il vaut bien mieux une réponse approximative à une question pertinente, souvent vague, qu'une réponse exacte à une question sans intérêt, qui peut toujours être rendue précise.

[15] Citation empruntée à [Jaynes95]



Khépéra vu de face



Khépéra vu de dessus

Figure 9

Prenons comme exemple le petit robot mobile Khépéra conçu par l'EPFL[16] et construit et commercialisée par K-team[17]. Khépéra est un robot mobile à deux roues, de 57 mm de diamètre et 29 mm de hauteur, pour un poids total de 80g (voir figure 9).

Il est équipé de 8 capteurs de lumière (6 devant et 2 derrière) prenant des valeurs entre 0 et 511 en fonction décroissante de l'intensité lumineuse (stockées dans les variables L_1 ... L_8, voir figure 10). Ces 8 capteurs peuvent aussi être exploités en tant que proximètres infrarouges prenant des valeurs entre 0 et 1023 en fonction décroissante de la distance[18] de l'obstacle (stockées dans les variables V_1 ... V_8, voir figure 10).

Il est commandé par les vitesses de ses roues gauche et droite (stockées dans les variables M_g et M_d).

L'espace sensori-moteur de ce robot est donc un espace de dimension 18 construit sur les variables L_1 ... L_8, V_1 ... V_8, M_g et M_d.

De ces variables sensorielles et motrices de base, on tire par prétraitement les 4 variables *Lum*, *Dir*, *Prox* et *Vrot*:

- *Lum* est une variable correspondant à peu près à la direction de plus grande illumination. *Lum* prend une valeur entre 0 et 7 correspondant au numéro (minoré de 1) de la cellule photoélectrique la plus éclairée.
- *Dir* est une variable correspondant très approximativement à la direction de l'obstacle le plus proche (voir figure 10). *Dir* prend des valeurs entières entre -10 (obstacle à gauche du robot) et 10 (obstacle à droite du robot). *Dir* est défini par la formule :

$$Dir = \mathrm{P}artieEnti\ re\left(\frac{-90V_1 - 45V_2 - 5V_3 + 5V_4 + 45V_5 + 90V_6}{9(V_1 + V_2 + V_3 + V_4 + V_5 + V_6) + 1}\right) \qquad [3.1]$$

[16] Ecole Polytechnique Fédérale de Lausanne

[17] http://www.K-team.com/

[18] En fait les valeurs de ces capteurs dépendent de plusieurs facteurs: la distance de l'objet, sa couleur, sa matière et la qualité de sa surface.

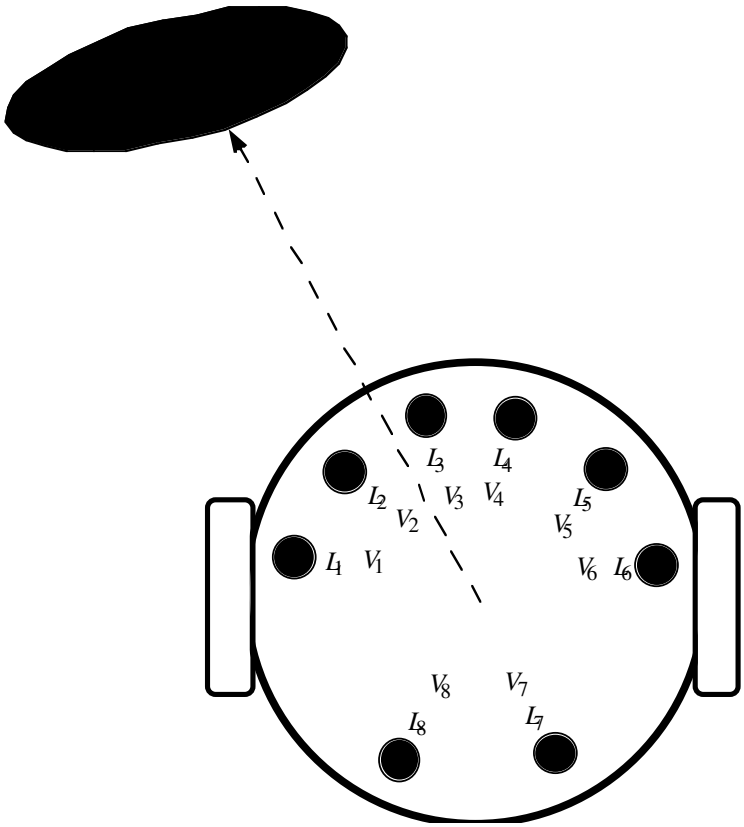


Figure 10 : les variables sensori-motrices de Khépéra

- *Prox* est une variable correspondant à peu près à la proximité de l'obstacle le plus proche (voir figure 10). *Prox* prend des valeurs entières entre 0 (obstacle très loin du robot) et 15 (obstacle très proche du robot). *Prox* est défini par la formule :

$$Prox = PartieEntiere\left(\max(V_1, V_2, V_3, V_4, V_5, V_6)/16\right) \quad [3.2]$$

- Le robot est piloté uniquement en rotation à vitesse de translation constante. Les ordres moteurs lui sont envoyés par la variable *Vrot* calculée comme la différence entre la vitesse de la roue gauche et celle de la roue droite. Elle prend des valeurs entières entre -10 (à fond vers la gauche) et +10 (à fond vers la droite).

Ces 4 variables, *Lum*, *Dir*, *Prox* et *Vrot* viennent s'ajouter aux 18 précédentes et l'on obtient un espace des phases de dimension 22.

Cet espace des phases est effectivement très structuré. Il existe d'innombrables et très fortes dépendances entre ces 22 variables. Par exemple :

- les valeurs des 8 variables Li sont a priori corrélées par le niveau moyen d'éclairage ;
- si le proximètre V3 a une valeur très élevée, il est probable que V2 et V4 aient aussi des valeurs élevées ;
- si Mg et Md sont tous 2 négatifs (le robot recule), il est prévisible que les valeurs de V2, V3, V4 et V5 vont diminuer puisque le robot va s'éloigner des obstacles frontaux ;
- si le robot avance et évite les obstacles alors il est probable qu'une valeur de *Dir* négative (obstacle à gauche) corresponde à une valeur de *Vrot* positive (le robot tourne vers la droite) et inversement ;
- etc..

A.3.2 Principe de l'approche F+D

Comme l'approche F+I, l'approche F+D repose sur l'emploi d'un système formel manipulant des signes et obéissant à des règles purement syntaxiques. Le système formel de F+D va être régi par les lois des probabilités (voir dans ce même numéro l'article « Interprétation ou Description (II) ») alors que, généralement, celui de F+I est régi par les lois de la logique. Par contre, les calculs formels ne sont plus rendus effectifs par le biais d'une *interprétation* reliant signes et objets de l'environnement, mais par le biais de *descriptions* codant les dépendances entre valeurs des variables de l'espace des phases.

Les signes manipulés par le système formel sont les valeurs des variables sensorielles, motrices et internes du robot. Ils n'ont pas pris individuellement de signification particulière. Pas plus que la couleur d'un pixel sur une rétine n'a de signification et ne permet de se faire une idée quelconque de l'image. Leurs seules contreparties sont des courants électriques dans des capteurs ou des actionneurs, ou des valeurs de variables informatiques. C'est collectivement, par leurs dépendances mutuelles, que les valeurs des variables de l'espace des phases prennent de l'intérêt. Ce sont ces dépendances que les *descriptions* vont coder sous forme de distributions de probabilités.

Une *description* est une notion mathématique tout aussi précisément définie que l'était le concept d'*interprétation*. Une *description* est définie comme une distribution de probabilités sur un sous-espace de dimension n de l'espace des phases (défini par le choix d'une base de n variables $X_1 \ldots X_n$). La définition d'aucun « domaine de discours », projetant par catégorisation une structure ensembliste sur le phénomène, n'est nécessaire. Les symboles manipulés ne sont plus associés à aucun référents distaux mais uniquement aux valeurs des variables proximales.

Une *description* est construite par un processus d'apprentissage sur un ensemble de données expérimentales D à partir d'un ensemble de connaissances préalables C données au système par le concepteur. Une *description* est donc dénotée formellement :

$$P(X_1 \ldots X_n \mid DC)^{19} \qquad [3.3]$$

On peut, si l'on connaît $P(X_1 \ldots X_n \mid DC)$, calculer n'importe quelle probabilité :

$$P\left(\left[X_i = x_i\right] \ldots \left[X_j = x_j\right] \mid \left[X_k = x_k\right] \ldots \left[X_l = x_l\right] DC\right)^{20} \qquad [3.4]$$

Ces calculs sont purement formels : comme pour l'approche F+I, à partir de suite de signes et de règles syntaxiques, on produit de nouvelles suites de signes. Cependant, alors que pour l'approche F+I, les règles syntaxiques sont, en général, inspirées par les règles de la logique, de l'algèbre ou de la géométrie, elles sont, pour l'approche F+D dérivées des probabilités.

Dans le cadre sensori-moteur, une *description* permet de rendre effectif ces calculs formels de manière originale. Supposons que S soit l'ensemble de variables sensorielles de la *description*, M l'ensemble des variables motrices et I l'ensemble des variables internes (ou

---

19 La barre verticale dénote le conditionnement. X1 ... Xn sont connues conditionnellement à D et C.

20 On note en majuscule les variables et en minuscule les valeurs de ces variables.

contextuelles). On peut alors, grâce à la *description*, traduire les résultats des calculs formels en :

- Commandes motrices : P(M | S I D C) permet de choisir les ordres moteurs pour atteindre une situation sensorielle S souhaitée connaissant le contexte I (voir § III.3.1).
- Prévisions sensorielles : P(S | M I D C) permet de prédire les conséquences sensorielles d'un jeu d'ordres moteurs dans un certain contexte I.
- Reconnaissance de contexte : P(I | S M D C) permet de reconnaître le contexte le plus probable connaissant une situation sensori-motrice (voir § III.3.3).

Ces calculs ont les remarquables propriétés suivantes :

- Ils restent parfaitement valables en situation d'incomplétude. Il est possible de choisir des commandes motrices, de faire des prévisions sensorielles et de reconnaître des situations même si seulement une partie des valeurs des variables de S, M et I sont connues. Par exemple :
  * P(M | S D C) permet de choisir des commandes motrices connaissant la situation sensorielle mais ignorant le contexte.
  * P(M | S1 S4 I D C) permet de choisir les commandes motrices connaissant seulement une partie de la situation sensorielle (par exemple parce que certains capteurs sont en panne).
  * P(I | M1 M3 S1 S4 D C) permet de trouver le contexte le plus probable en ne connaissant qu'une partie de la situation sensori-motrice.
  * Etc.
- Ils permettent de propager les incertitudes. Il se peut que certaines valeurs des variables de l'espace des phases ne soient pas connues précisément, soit que le capteur ou l'actionneur correspondant fournisse des valeurs entachées d'incertitude, soit que ces valeurs doivent, elles-mêmes, être inférées par un raisonnement probabiliste. A ces variables, correspond alors une distribution de probabilité. Ces distributions peuvent intégralement être prises en compte dans le raisonnement probabiliste (voir § III.3.2).

A.3.3 L'approche F+D mise en pratique

Dans ce paragraphe, nous décrivons quelques-unes des expériences menées avec le Khépéra afin d'illustrer le fonctionnement et les capacités de l'approche F+D

A.3.3.1 Utilisation et apprentissage de comportements réactifs

Dans cette première série d'expériences, nous avons cherché à doter Khépéra de comportements purement réactifs vis-à-vis des obstacles. Les commandes motrices sont directement inférées des valeurs sensorielles.

Prenons l'exemple de l'évitement d'obstacle. Le but poursuivi est de donner la capacité à Khépéra d'éviter les obstacles et d'aller en ligne droite lorsqu'il est en espace libre. Nous avons choisi d'utiliser pour cette expérience les 3 variables *Dir*, *Prox* et *Vrot*. La *description* correspondante va donc être une distribution de la forme :

$$P(DirProxVrot \mid DC) \qquad [3.5]$$

Le protocole expérimental adopté est constitué de 2 phases:

- Une phase d'apprentissage pendant laquelle Khépéra est piloté par un opérateur à l'aide d'un « joystick ». Durant cette phase, les données expérimentales D (valeurs des variables sensorielles *Dir* et *Prox* et de la commande motrice *Vrot*) sont enregistrées et

les paramètres, permettant de stocker P(*Dir Prox Vrot* | D C) en machine, sont identifiés
- Une phase de restitution où Khépéra évite les obstacles tout seul. Les calculs formels probabilistes sont alors utilisés pour calculer à chaque instant la commande motrice *Vrot* à envoyer aux roues en fonction des données sensorielles *Dir* et *Prox* provenant des capteurs de proximité.

La *description* P(*Dir Prox Vrot* | D C) obtenue lors de l'apprentissage dépend évidemment des expériences D, mais aussi, et d'une certaine manière : surtout, des connaissances préalables C.

Les connaissances préalables C données par le concepteur peuvent être regroupées en 4 catégories de natures différentes :
- Les « connaissances préalables structurelles » sont définies par l'ensemble des variables X_1 ... X_n sur lesquelles la distribution de probabilité va être construite. Pour notre exemple, comme il a déjà été dit, 3 variables ont été choisies : *Dir*, *Prox* et *Vrot*.
- Les « connaissances préalables de dépendances » sont données par une structure de dépendance[21] permettant de décomposer de manière unique une *description* comme un produit de distributions plus simples. Ces simplifications essentielles sont fondées sur des considérations d'indépendance conditionnelle entre variables. Pour l'exemple qui nous occupe, les règles des probabilités donnent :

$$P(DirProxVrot \mid DC) = P(Dir \mid DC)P(Prox \mid DirDC)P(Vrot \mid ProxDirDC) \quad [3.6]$$

Nous pensons a priori qu'il n'existe pas de corrélation particulière entre la direction dans laquelle on observe un obstacle et la distance à cet obstacle. Nous traduisons cette connaissance pour notre robot en lui disant que *Prox* est indépendante de *Dir* conditionnellement à D et C.

$$P(Prox \mid DirDC) = P(Prox \mid DC) \quad [3.7]$$

On obtient donc :

$$P(DirProxVrot \mid DC) = P(Dir \mid DC)P(Prox \mid DC)P(Vrot \mid DirProxDC) \quad [3.8]$$

- Les « connaissances a priori » donnent des valeurs initiales (avant apprentissage) aux distributions ou explicitent une méthode pour les calculer à partir d'autres distributions connues. Une connaissance a priori peut aussi, par exemple, exprimer qu'une distribution est supposée ne pas évoluer en fonction des données.
  Nous n'avons aucune raison de penser que le robot sera amené à rencontrer les obstacles suivant une direction privilégiée ou à une distance particulière. Nous adoptons donc comme a priori que tous les couples (*Dir, Prox*) sont équiprobables, ce qui se traduit par :

$$P(Dir \mid DC) = P(Dir \mid C) = \frac{1}{21} \quad [3.9]$$

et par :

$$P(Prox \mid DC) = P(Prox \mid C) = \frac{1}{16} \quad [3.10]$$

[21] Une structure de dépendance est une généralisation de la notion de réseaux bayésiens due à J. Pearl [Pearl91].

- Enfin, les « connaissances préalables d'observation », qui se décomposent en la spécification d'une représentation paramétrique et d'un processus d'apprentissage (identification des paramètres) pour chacune des distributions de la structure de dépendances. Une représentation paramétrique permet d'une part, de représenter ces distributions en machine et d'autre part, d'identifier leurs paramètres par apprentissage.
  Pour cette expérience, on a utilisé une loi de succession de Laplace comme représentation paramétrique de :

$$P([Vrot = v] \mid [Dir = d][Prox = p]DC) = \frac{m_v + 1}{m_{dp} + 21} \qquad [3.11]$$

  Où :
  * m_{dp} est le nombre de fois où la situation sensorielle *Dir*=*d* et *Prox*=p a été rencontrée pendant l'apprentissage
  * m_V est le nombre de fois ou *Vrot*=v a été observé pour la situation sensorielle précédente.
  * 21 est le nombre de valeurs possibles pour *Vrot*.

  Pour un nombre important d'expériences, la loi de succession de Laplace se rapproche d'un simple histogramme défini par m_v/m_{dp}. Les termes correctifs 1 et 21 servent à traiter les cas où le nombre d'expériences est faible. En particulier pour $m_v=m_{dp}=0$, on trouve 1/21 soit la distribution uniforme.

Par le calcul formel on peut alors, en phase de restitution, commander Khépéra en choisissant à chaque instant *Vrot* suivant la distribution P(*Vrot* | *Dir Prox* D C). La commande de Khépéra par cette méthode fonctionne très bien. Après un temps d'apprentissage de quelques dizaines de secondes, on obtient le comportement souhaité (voir figure 11). Le comportement obtenu s'avère être très « robuste » au changement de position, de taille, de forme, de matière, de couleurs[22] et même de vitesse des obstacles, ainsi qu'aux conditions d'éclairage de la scène.

Avec la même méthode et les mêmes connaissances préalables, en changeant l'apprentissage téléopéré au « joystick », on a appris a Khépéra à suivre les murs et contours (voir figure 12), à pousser des objets et à poursuivre les obstacles mobiles. Avec la même méthode, sur la base des variables *Vrot* et *Lum,* nous avons appris à Khépéra un comportement de phototaxie représenté par la *description* P(*Vrot Lum* | D' C'). [23]

[22] En fait, seul les obstacles noirs et mats, que les proximètres ne « voient » pas du tout, posent un problème.

[23] Voir les films de démonstration correspondant à tous ces comportements à l'adresse :

http://www-leibniz.imag.fr/LAPLACE/Demonstration/Robotique/Surveillance

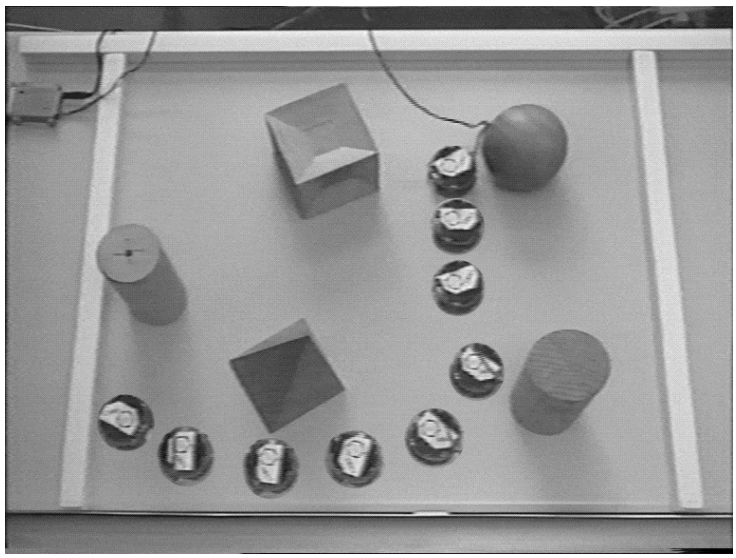


Figure 11 : comportement d'évitement d'obstacles

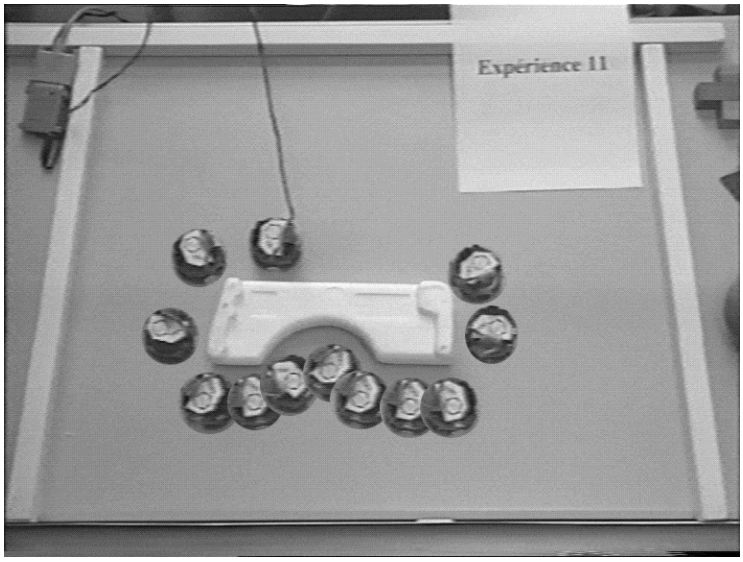


Figure 12 : comportement de suivi de contour

A.3.3.2 Composition simple de comportements réactifs

Un but souvent poursuivi et rarement atteint par les approches réactives de la robotique est de pouvoir composer entre eux des comportements appris séparément. Par exemple, ici, nous aimerions obtenir de Khépéra qu'il aille vers la lumière en évitant les obstacles à partir des comportements de phototaxie et d'évitement d'obstacles déjà appris.

L'approche F+D à l'avantage de permettre de poser très clairement ce genre problème:
- Nous cherchons P(*Vrot* | *Lum Dir Prox* D C)
- Nous avons P(*Vrot* | *Dir Prox* D1 C1) et P(*Vrot* | *Lum* D2 C2)
- Nous voulons que (par exemple):
    * en l'absence de données sur la lumière, Khépéra se contente d'éviter les obstacles :

$$P(Vrot \mid DirProxC) = P\left(Vrot \mid DirProxD_1C_1\right) \qquad [3.12]$$

* En l'absence de données sur les obstacles, Khépéra se contente d'aller vers la lumière :

$$P(Vrot \mid LumC) = P(Vrot \mid LumD_2C_2) \quad [3.13]$$

* En l'absence d'obstacle, Khépéra aille vers la lumière :

$$P(Vrot \mid Lum[Prox = 0]C) = P(Vrot \mid LumD_2C_2) \quad [3.14]$$

* A grande proximité d'un obstacle, Khépéra évite coûte que coûte ceux-ci :

$$P(Vrot \mid LumDir[Prox = 15]C) = P(Vrot \mid DirProxD_1C_1) \quad [3.15]$$

On obtient ainsi une série de contraintes pesant sur la forme paramétrique recherchée de P(*Vrot* | *Lum Dir Prox* D C). Ce genre de problème est en général largement sous-contraint, laissant de nombreuses possibilités ouvertes pour le choix final de P(*Vrot* | *Lum Dir Prox* D C).

Ici, par exemple, les règles des probabilités nous permettent d'introduire une variable *H* pouvant prendre 2 valeurs et d'écrire:

$$\begin{aligned} P(Vrot \mid DirProxLumDC) = {} & P(Vrot \mid DirProxLum[H = 1]DC)P([H = 1] \mid DirProxLumDC) \\ & + P(Vrot \mid DirProxLum[H = 2]DC)P([H = 2] \mid DirProxLumDC) \end{aligned} \quad [3.16]$$

Si l'on considère cette variable comme une manière de choisir entre le comportement de phototaxie et le comportement d'évitement, on peut poser :

$$P(Vrot \mid DirProxLum[H = 1]DC) = P(Vrot \mid DirProxD_1C_1) \quad [3.17]$$

et

$$P(Vrot \mid DirProxLum[H = 2]DC) = P(Vrot \mid LumD_2C_2) \quad [3.18]$$

Finalement, en posant *a priori* :

$$P([H = 1] \mid DirProxLumDC) = P([H = 1] \mid ProxC) = \frac{Prox}{15} \quad [3.19]$$

On obtient une solution qui satisfait toutes les contraintes ([3.12], [3.13], [3.14] et [3.15]) posées.

En pratique, on obtient ainsi, sans aucun apprentissage supplémentaire, un comportement de phototaxie avec évitement d'obstacle (voir figure 13[24]).

---

24 Voir le film de démonstration correspondant à ce comportement à l'adresse :

http://www-leibniz.imag.fr/LAPLACE/Demonstration/Robotique/Surveillance

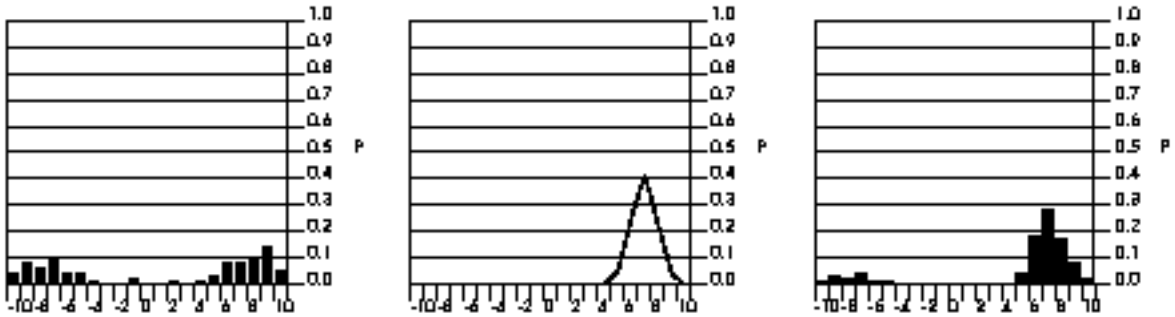


Figure 13 : composition simple de comportements réactifs
Les ordonnées représentent les probabilités P(Vrot | ...)dans diverses situations :
a : au départ, face à un obstacle à moyenne distance.
Khépéra n'a pas de réelle préférence entre la droite et la gauche
b : comportement de phototaxie, cas où la source lumineuse est à droite.
c : comportement mixte obtenu par la méthode de composition expliquée ci-dessus :
si la source lumineuse est à droite Khépéra préfère éviter l'obstacle par la droite.

A.3.3.3 Reconnaissance d'objet

Il est bien évident que des comportements un tant soit peu complexe ne peuvent être purement réactifs. En particulier, il est souvent nécessaire de prendre des décisions sur une base de temps plus large que le simple instant présent et sur un domaine spatial plus grand que la pure information sensorielle immédiate.

Dans cette expérience, grâce à l'approche F+D, nous faisons un premier pas dans cette direction. L'expérience va consister pour Khépéra à apprendre à reconnaître un certain nombre d'objets. Cet apprentissage est totalement non supervisé. On présente les objets sans dire au robot à quel objet il a à faire, ni même s'il s'agit d'un nouvel objet ou d'un objet déjà connu. Khépéra doit donc les classer de lui-même, et créer, de lui-même, une nouvelle classe quand un nouvel objet lui est présenté.

Le protocole expérimental est le suivant :

- Khépéra est placé en espace libre face à un objet. Il avance en ligne droite jusqu'à ce que ses proximètres détectent l'objet puis il commence à suivre les bords de cet objet grâce au comportement de « suivi de contours » décrit précédemment (voir figure 12).
- Pendant qu'il effectue ainsi le tour de l'objet, Khépéra enregistre les commandes *Vrot* qu'il envoie à ses moteurs. Il arrête cet enregistrement lorsque la somme algébrique des commandes *Vrot* égale 360° (Ce qui correspond en pratique à une estimation plutôt grossière de la fin du tour). L'enregistrement des valeurs de *Vrot* est lissé par une moyenne mobile sur 50 valeurs successives. Cette série de données lissées est utilisée pour calculer 4 nouvelles variables qui vont servir à discriminer les objets :
  * *Nvd* « Nombre de Virages à Droite » : Un « virage à droite » est une suite d'au moins 30 valeurs négatives de *Vrot* lissées.
  * *Nvg* « Nombre de Virages à Gauche ».
  * *Per* « Périmètre » : Nombre total de valeurs de *Vrot* enregistrées en un « tour ».
  * *Lgc* « Longueur du plus Grand Coté » : Plus grand nombre de valeurs de *Vrot* entre 2 « virages ».

  Par exemple, pour un « tour » effectué autour de l'objet de la figure 12 on obtient : *Nvd* = 1, *Nvg* = 4, *Per* = 710, *Lgc* = 173.
- A partir des valeurs ainsi obtenues pour ces 4 variables, Khépéra, par inférence probabiliste, catégorise l'objet soit dans une classe correspondant au prototype d'un

objet déjà rencontré (qui est alors enrichi par cette nouvelle occurrence) soit dans une nouvelle classe qu'il crée parce qu'il pense être en présence d'un objet nouveau.

Les variables manipulées sont les suivantes : *Nvd*, *Nvg*, *Per*, *Lgc* et *O*. *O* est la variable permettant d'identifier les classes d'objets. Lorsque n classes d'objets ont déjà été identifiées, *O* peut prendre n+1 valeurs différentes (la valeur *O*=0 est réservée à une classe de référence « neutre » et est utilisée pour détecter la nouveautée). La *description* qui nous intéresse est donc :

$$P(NvdNvgPerLgcO \mid DC) \qquad [3.20]$$

On suppose que *Nvg*, *Nvd*, *Per* et *Lgc* sont des variables indépendantes conditionnellement à *O*. On obtient donc comme structure de dépendance :

$$P(NvdNvgPerLgcO \mid DC) = P(O \mid DC)P(Nvd \mid ODC)P(Nvg \mid ODC)P(Per \mid ODC)P(Lgc \mid ODC) \qquad [3.21]$$

Comme connaissances préalables d'observation, on choisit des gaussiennes comme représentation paramétrique des 4 distributions à identifier. Pour chaque valeur de *O* on identifie donc une moyenne et un écart type pour *Nvg*, *Nvd*, *Per* et *Lgc*.

La valeur *O*=0 est traitée à part. $P(NvdNvgPerLgc \mid [O=0]DC)$ est fixée *a priori* uniforme. Cette distribution va être utilisée comme référence pour détecter la nouveauté.

Pour répondre à la question qui lui est posée, Khépéra calcule pour chaque valeur possible de *O*:

$$P(O \mid NvdNvgPerLgcDC) = \frac{P(O \mid DC)P(Nvd \mid ODC)P(Nvg \mid ODC)P(Per \mid ODC)P(Lgc \mid ODC)}{\sum_{O=0}^{O=n+1} P(O \mid DC)P(Nvd \mid ODC)P(Nvg \mid ODC)P(Per \mid ODC)P(Lgc \mid ODC)} \qquad [3.22]$$

Il trouve ainsi la valeur de *O* la plus probable. Si cette valeur est i≠0, Khépéra considère avoir reconnu l'objet i et modifie les paramètres correspondants pour tenir compte de cette nouvelle instance. Si cette valeur est 0, Khépéra considère être en face d'un nouvel objet. Il crée une nouvelle classe (nouvelle valeur possible pour *O*) sur la base des données qu'il vient de recueillir.

O est une variable plus abstraite et plus synthétique que les variables sensori-motrices. *O* a une constante de temps d'acquisition plus longue que les variables sensori-motrices puisqu'elle suppose l'intégration dans l'espace et dans le temps d'un certain nombre de valeurs des ces variables de bases. *O* peut à son tour être utilisé dans de nouvelles *descriptions*. On peut espérer ainsi construire de véritables hiérarchies de *descriptions*, de plus en plus abstraites, correspondant à des comportements de plus en plus complexes, avec des constantes de temps de plus en plus longues. Nous en verrons un exemple à l'alinéa suivant.

La figure 15 montre les objets qui ont été appris avec cette méthode. Chacun a été présenté 4 fois, en suivant un ordre aléatoire. Seuls les 2 objets à base carrée, en haut à droite de la photo, ont été confondus. Ils sont en fait parfaitement indiscernables avec les 4 variables choisies. Dans tous les autres cas, les objets ont été reconnus sans erreur. La nouveauté a, elle aussi, été identifiée à chaque fois sans erreur et sans oubli.

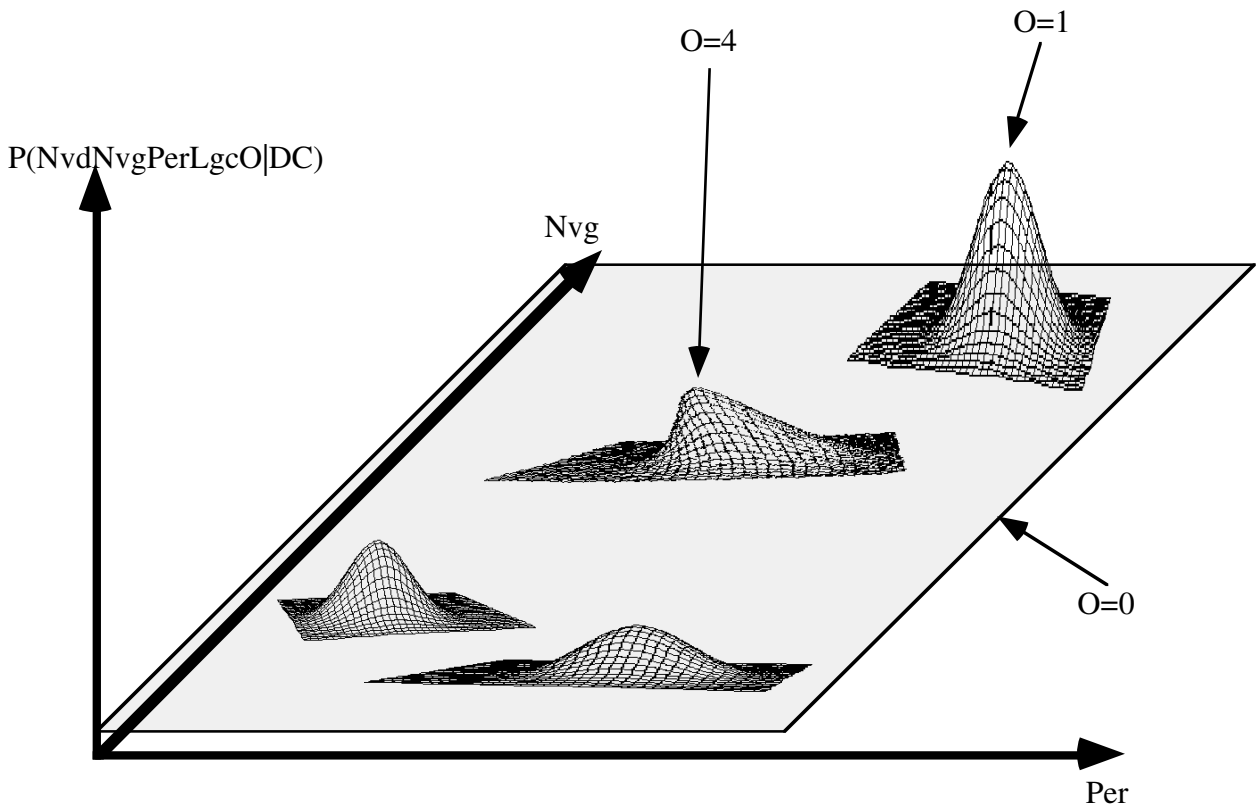


Figure 14 : $P(NvdNvgPerLgcO \mid DC)$

Seules les dimensions Per et Nvg sont représentées. Chaque pic correspond à un objet appris. La distribution uniforme (plan grisé) sert de référence pour reconnaître la nouveauté. Si pour un jeu de valeurs de Nvg, Nvd, Per et Lgc, c'est la distribution uniforme qui est la plus probable, Khépéra pense avoir rencontré un nouvel objet.

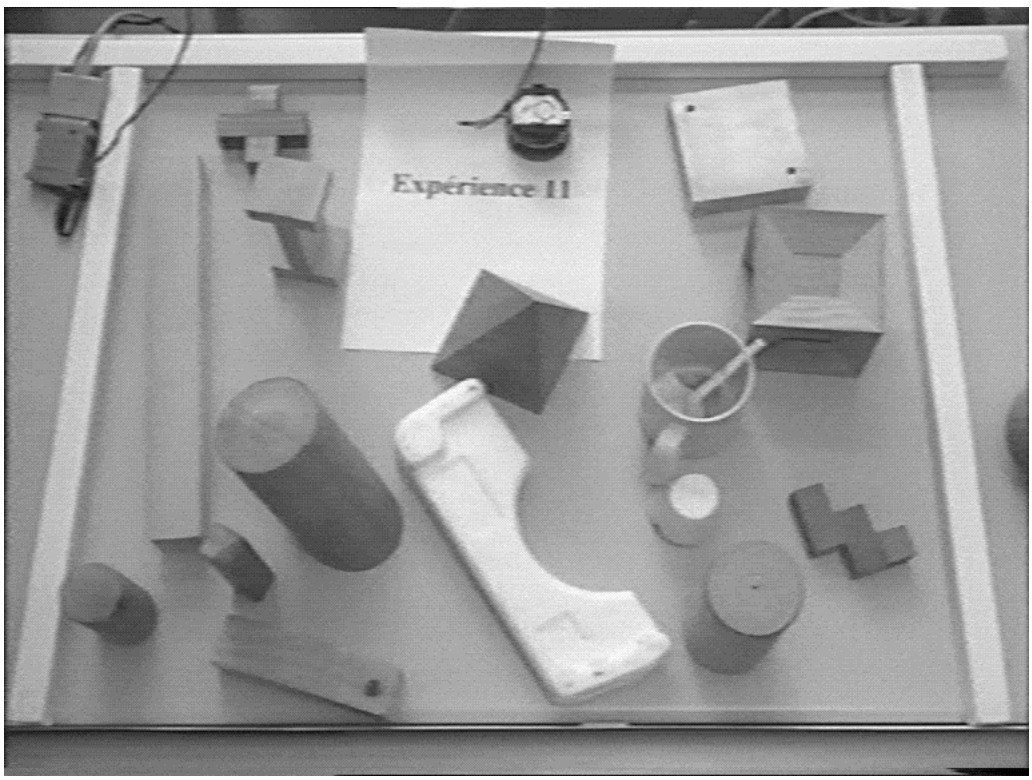


Figure 15 : Les objets présentés à Khépéra

A.3.3.4 Une tâche de surveillance

Enfin, et surtout, à l'aide de tous les comportements précédents, préalablement acquis, nous avons programmé et « dressé » Khépéra pour une tâche de surveillance beaucoup plus complexe. Dans cette tâche, Khépéra patrouille un environnement inconnu, donne l'alarme s'il détecte des mouvements, éteint (souffle) un incendie (une bougie) s'il en rencontre un (voir figure 16), identifie les objets qu'il côtoie et rentre seul à sa base quand on lui en donne l'ordre ou quand ses réserves d'énergie baissent.

Outre les capteurs et actionneurs précédemment décrits, on utilise pour cette tâche une caméra linéaire à 64 pixels (pour la détection du mouvement et des bougies), les odomètres (qui permettent de savoir quand le tour d'un objet a été effectué) et une micro-turbine (pour souffler les bougies).

Cette tâche utilise 5 comportement réactifs acquis par apprentissage qui constituent les types de déplacements de bases à sa disposition :
- l'évitement d'obstacle (voir alinéa III.3.1) ;
- le suivi de contour (voir alinéa III.3.1) ;
- l'évitement d'obstacle en allant vers la lumière (voir alinéa III.3.2) ;
- le comportement de pompier ;
- l'immobilité.

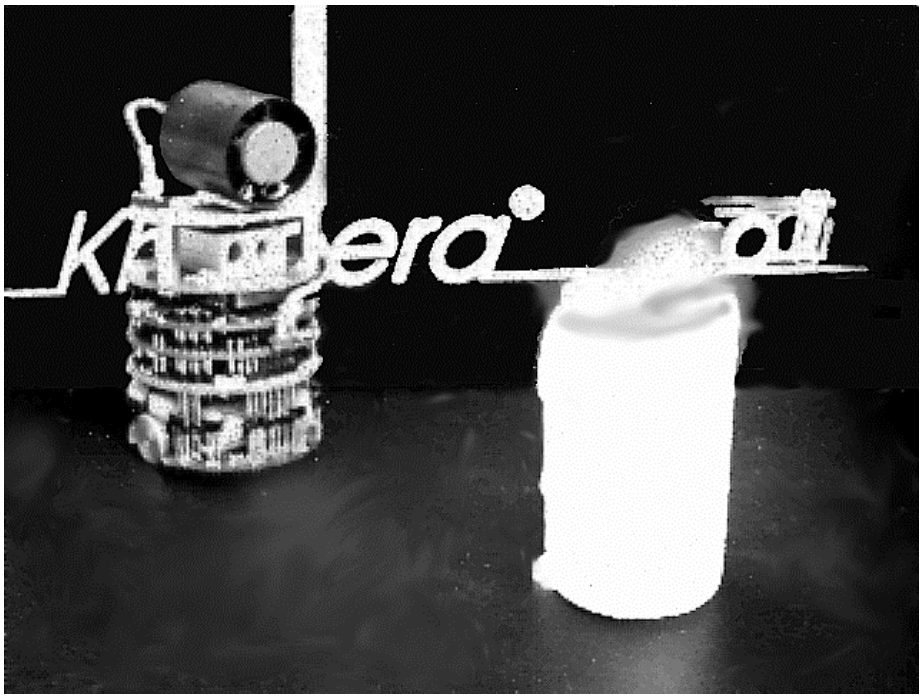


Figure 16 : Khépéra soufflant une bougie

Voir aussi le film complet à l'adresse
http://www-leibniz.imag.fr/LAPLACE/Demonstration/Robotique/Surveillance

Ces cinqs types de déplacement élémentaires sont combinés les uns avec les autres, par inférence probabiliste, pour obtenir 6 comportements plus complexes :
- la patrouille où Khépéra se déplace aléatoirement dans son environnement en s'arrêtant de temps en temps pour vérifier que rien ne bouge en dehors de lui ;

- le traitement de l'incendie où Khépéra se positionne par rapport à la bougie et la souffle ;
- la reconnaissance d'objet (voir alinéa III.3.3) ;
- le retour à la base sur ordre ;
- le retour à la base par manque d'énergie ;
- la recharge de son énergie, immobile, à la base.

A.3.4 Les réponses apportées par l'approche F+D aux difficultés de l'approche F+I

L'approche F+D tranche le nœud gordien de la catégorisation.

Elle ne manipule pas de symbole référant aux objets et aux classes d'une hypothétique réalité supposée objective. Elle manipule uniquement des valeurs de variables, dont la contrepartie est proximale : un courant électrique dans un capteur ou la valeur d'une variable informatique. Aucun lien n'est à fournir entre ces marques et les objets ou classes d'objets de l'environnement. Plus besoin de justifier ces deux postulats si contestables :

- Le robot est capable, tout comme son concepteur, chacun à travers son appareil sensori-moteur, de percevoir une STRUCTURE ENSEMBLISTE (objets et ensemble d'objets) de l'environnement.
- Le robot et son concepteur sont tous deux capables d'identifier les MEMES objets et ensembles d'objets.

L'analogie pertinente pour l'approche F+I est celle de la langue. L'analogie pertinente pour l'approche F+D est celle de l'image. L'environnement n'est pas appréhendé par des mots, forcément réducteurs, sinon caricaturaux, mais par les *descriptions*, de véritables « images » où les probabilités tiennent lieu de « niveaux de gris » (voir figure 14). L'inférence n'est pas une manipulation linguistique contraignante, mais peut être vue comme l'application d'un ensemble de filtres et d'opérateurs de composition permettant de combiner directement les unes avec les autres ces différentes « images mentales » probabilistes.

Reste la raison fondamentale invoquée pour contester ces postulats, à savoir l'inévitable incomplétude d'un système formel pour modéliser un environnement physique. Le système formel probabiliste de l'approche F+D n'échappe pas plus à cette incomplétude que les systèmes formels logiques des approches F+I, mais il le reconnaît et sait raisonner au mieux malgré cette incomplétude. Pour cela, l'approche F+D renonce explicitement à construire des modèles complets. Elle postule, au contraire, que l'observateur, avec ses connaissances préalables et ses a priori, doit être impérativement pris en compte dans toute tentative d'automatisation de la perception, de la prise de décision ou de l'action. Elle ne cherche donc pas à construire de modèles d'une hypothétique réalité mais toujours à élaborer par l'expérience des *descriptions* d'une interaction entre un système sensori-moteur et son environnement ou, autrement dit, du point de vue très « subjectif » que ce système peut avoir de son environnement. Adopter ce point de vue subjectif de l'observateur c'est reconnaître la nature par essence incomplète de la connaissance du phénomène et donc accepter qu'il subsiste irrémédiablement des sources d'incertitude. *L'apprentissage et les techniques de maximum d'entropie permettent de transformer l'incomplétude (ce qui n'a pas été explicitement modélisé) en incertitude (les « descriptions »). L'inférence probabiliste prend le relais ensuite et permet de raisonner avec cette incertitude.* On pourra lire dans ce même numéro l'article intitulé « Interprétation ou Description (II) : *Fondements mathématiques de l'approche F+D »* pour une présentation approfondie de l'inférence probabiliste et du principe de maximum d'entropie.

Finalement, l'approche F+D répond bien aux objectifs qu'on lui avait fixés en donnant des réponses approximatives aux questions jugées pertinentes, par essence vagues, et en récusant qu'une réponse exacte puisse être apportée à autre chose qu'une question trop formelle pour être sensori-motrice.

A.3.5 Notes bibliographiques

L'idée de base de l'approche F+D est à relier à de nombreux travaux de sciences cognitives qui, chacun avec leur propre terminologie et avec de nombreuses nuances, nient la possibilité de modéliser l'environnement par une simple opération de type “traitement de l'information” et défendent la thèse que toute activité cognitive sensori-motrice doit être « enracinée » dans une interaction physique effective avec l'environnement.

Ces travaux vont du concept de « corporéité » de Merleau-Ponty (Merleau-Ponty42), (Merleau-Ponty45) à la notion d'« enaction » de Varela (Varela89a), (Varela89b), (Varela91), (Varela93) en passant par la « psychologie écologique » de Gibson et sa notion d'« affordance » (Gibson79), par le concept d'« action en situation » de Suchman (Suchman87) et finalement par les travaux de Braitenberg (Braitenberg83) et Brooks (Brooks91b).

De nombreuses logiques « alternatives » ont été proposées pour essayer de pallier les difficultés de l'approche F+I et pour gérer l'incertitude et l'incomplétude de l'information venant d'un phénomène physique et naturel. Pour aborder le vaste sujet que constituent ces logiques « alternatives », se référer à (Rosser58) et (Turner84) pour les logiques multivaluées, à (Bradley79), (McCarthy80), (McDermott80), (McDermott82), et (Turner84) pour les logiques modales et non monotones, à (Zadeh65), (Goguen67) et (Dubois80) pour la logique floue, à (Dempster68) et (Shafer76) pour la théorie dite de « Dempster-Shafer » et enfin, à (Zadeh78) et (Dubois85) pour la théorie des possibilités.

Des débats passionnés ont eu lieu autour des mérites relatifs des diverses théories du raisonnement plausible. On trouvera des arguments en faveur des probabilités dans (Cox61), (deFinetti72), (Cheeseman85), (Robert91) et (Pearl91).

Chercher à élaborer des structures internes rendant compte de dépendances entre variables observées est un paradigme déjà largement exploré et connu sous le nom générique d'apprentissage. Il en est ainsi, par exemple, de la théorie de l'identification et des très nombreux et très importants travaux qui y sont liés. Elle cherche, étant donné un modèle mathématique d'un phénomène, à identifier la forme et la valeur des paramètres de ce modèle pour décrire « correctement » un ensemble d'observations. On distingue l'identification paramétrique et l'identification non paramétrique. La première impose des contraintes de type analytique sur le modèle (notamment contraintes de « lissage » et de dérivabilité sur les fonctions) et cherche dans une classe de fonctions laquelle peut le mieux rendre compte des données. La seconde, plus restrictive, part d'une forme fonctionnelle donnée a priori et cherche uniquement à l'ajuster aux données. On distingue aussi la théorie de l'approximation et la théorie de l'estimation. La première prend en compte explicitement la gestion de l'incertitude, alors que la seconde assume qu'une *description* analytique sans incertitude est suffisante. Les applications de l'identification sont très nombreuses. Contentons-nous de citer deux ouvrages introductifs de qualité, (Tarentola87) plutôt théorique, et (Richallet91) couvrant la pratique de ces techniques, ainsi que les thèses de A. Labbi (Labbi93) et R. Balaniuk (Balaniuk96) qui ont l'avantage de faire le lien entre les réseaux neuronaux et l'identification.

L'approche par les réseaux de neurones formels peut être vue comme un cas particulier de l'identification. La forme fonctionnelle donnée a priori correspond à l'architecture du réseau et les paramètres à identifier pour ajuster le modèle aux observations sont alors appelés « poids ». Les réseaux de neurones formels sont fréquemment utilisés en robotique pour faire de l'apprentissage. De nombreuses voies sont en cours d'exploration, certaines sont assez proches de celle préconisée dans cette étude. De plus, la tendance actuelle en théorie des réseaux de neurones formels est de montrer que ces réseaux sont des implantations plus ou moins astucieuses et efficaces d'estimateurs probabilistes. La théorie du raisonnement probabiliste et les techniques de maximum d'entropie introduites précédemment et qui sont développées dans l'article de ce même numéro « Interprétation ou Description (II) : Fondements mathématiques de l'approche F+D », apparaissent donc de plus en plus comme le support théorique des travaux effectués dans le domaine des réseaux de neurones formels. On peut d'une part espérer que l'existence d'une théorie mathématique claire des réseaux va pouvoir faire progresser le développement de ceux-ci et, d'autre part, que les algorithmes étudiés pour les réseaux vont constituer des implantations intéressantes (notamment parallèles) des méthodes théoriques proposées par le raisonnement probabiliste. Pour les liens théoriques entre réseaux de neurones formels et probabilités on pourra lire, entre autres, (Geman84), (Smolensky86), (Cotrell88), (Robert90b), (Bessière90), (Hervé90), (Robert91) et (Geman92). Pour les liens théoriques entre réseaux de neurones formels, probabilités et physique statistique on pourra lire aussi, par exemple, (Gardner89a), (Gardner89b), (Brunel92), (Grassberger92), (Nadal92), (Nadal93) et (Gutfreund94).

Ni la théorie de l'identification (approximation + estimation) ni celle des réseaux ne répondent cependant aux exigences que nous nous sommes fixées car, si elles permettent de construire des structures de données rendant compte des dépendances entre variables observées, elles ne permettent plus, comme c'était le cas dans l'approche F+I, de combiner entre elles ces structures pour dérouler un calcul formel. Tel est l'essence de la critique adressée par Fodor et Pylyshyn au « connexionisme » quand ils lui reprochent de ne plus pouvoir satisfaire les contraintes de « productivity », « systematicity » et « compositionality » qu'ils considèrent comme indispensables pour tout système cognitif. Concernant ces objections, l'article de référence est (Fodor88). On consultera les deux livres de Fodor (Fodor76) et (Fodor84) pour mieux comprendre cet article en le replaçant dans l'ensemble de la théorie cognitive qu'il préconise.

A.4 Une perspective en guise de conclusion : l'évolution contingente des connaissances préalables

La logique et l'approche F+I sont la théorie fondatrice et la mécanisation du courant « symbolique » ou « cognitiviste » qui suppose que la pensée a comme prérequis le discours. L'apport fondamental des probabilités et de l'approche F+D est de proposer une théorie mathématique fondatrice et un moyen de mécaniser l'alternative « sub-symbolique» ou « antéprédicative » qui suggère que la pensée puisse préexister au discours. Ces deux options épistémologiques fondamentales, qui se sont opposées sous diverses formes dans toute l'histoire de la philosophie et de la science, sont ainsi remises sur un pied d'égalité, quant à la possibilité de les analyser mathématiquement et de les expérimenter informatiquement, possibilité réservée jusque-là au seul paradigme cognitiviste.

C'est dire l'ambition de l'inférence probabiliste et le nombre de perspectives ouvertes pour poursuivre le travail décrit ici, qui peut être vu comme l'amorce d'un très vaste programme de recherche. Plutôt que d'énumérer trop rapidement une trop longue liste de travaux en cours et

de perspectives possibles, nous préférons en détailler une qui nous paraît à la fois fondamentale et abordable : l'évolution contingente des connaissances préalables.

L'approche F+D est un moyen de surmonter nombre des difficultés que l'approche F+I rencontre dans le cadre sensori-moteur. Cependant, il incombe au concepteur du système sensori-moteur une tâche absolument essentielle, le choix des connaissances préalables, qui conditionne entièrement le succès de l'entreprise. Sur le chemin vers « l'autonomie[25] » des systèmes sensori-moteurs, l'étape suivante est donc de se demander s'il est possible d'automatiser au moins partiellement cette tâche de sélection des connaissances préalables. Nous pensons que la formalisation probabiliste rigoureuse et unifiée de l'inférence et de l'apprentissage que propose l'approche F+D permet de poser clairement cette question et d'envisager les voies à suivre pour la résoudre. L'objectif de cette conclusion est de les présenter.

A.4.1 Le rôle du concepteur dans l'approche F+D

Dans l'approche F+D des systèmes sensori-moteurs, le rôle du concepteur est clairement identifié, il doit :

1 - concevoir la structure mécanique et électronique du robot ;
2 - spécifier les connaissances préalables (structurelles, de dépendance, a priori et d'observation) ;
3 - servir de moniteur au robot en le plaçant dans des situations qu'il juge pertinentes et éventuellement superviser son apprentissage, soit en lui donnant des exemples de comportements (par exemple, pour Khépéra en le pilotant avec le « joystick »), soit en lui fournissant une mesure de qualité sur ses comportements (apprentissage par renforcement).

Concernant le premier point, pour des raisons pratiques, il nous semble difficile pour l'instant de modifier automatiquement la structure physique et électronique du robot.

Le troisième point est très intéressant, mais n'est pas le sujet de cette conclusion où nous nous concentrerons sur le deuxième problème moins classique : comment rendre (partiellement) automatique la recherche des connaissances préalables.

A.4.2 La notion de contingence

Les ailes n'ont pas été « conçues » pour que les animaux puissent voler. Chez les insectes, elles furent sans doute, à l'origine, des organes de régulation de la température (voir Gould91, chapitre 6). Il s'avéra alors que la structure de ces organes donnait des capacités pour planer. Cet « effet de bord » apportant un avantage sélectif, il fut préservé et continua à évoluer pour devenir la fonction principale et conduire aux ailes que nous observons sur les insectes actuels.

[25] Du grec *autos*, soi-même et *nomos*, loi : qui se donne ses propres lois.

De nombreuses structures, de la biologie à la technologie (par exemple, la disposition des touches sur les claviers d'ordinateurs), ne peuvent s'expliquer de manière satisfaisante que si on les replace dans une telle perspective historique et évolutionniste. Elles sont incompréhensibles si l'on suppose qu'elles ont été « conçues » à partir d'une « tabula rasa » en vue de remplir leurs fonctions actuelles.

Cette notion appelée « contingence » est considérée par certains (voir en particulier l'œuvre de Stephen Gould (Gould85), (Gould89) et (Gould91)) comme un aspect essentiel de l'évolution des espèces animales. Elle n'a jamais, à notre connaissance, été appliquée à la construction des robots. Cela se comprend aisément, dans la mesure où la plupart des robots existant ont été « conçus ». Dans la perspective d'autonomie où nous nous plaçons dans ce paragraphe, il ne s'agit plus de concevoir les robots mais de leur donner les moyens de se « développer ». C'est alors que nous pensons que la contingence devient un facteur capital.

Comme nous l'avons vu tout au long de ce papier, l'incomplétude des modèles est la difficulté majeure que rencontre la robotique. Du « bruit » aux « aberrations » en passant par les « hasards » et les « accidents », tout ce qui est « imprévu », c'est-à-dire tout ce qui sort des limites étriquées de la modélisation est considéré comme parasite et nuisible.

L'approche F+D, comme nous l'avons vu, apporte un premier type de réponse à l'incomplétude : les « variables cachées », ce qui n'a pas été modélisé, engendrent de l'incertitude qui est prise en compte dans les *descriptions* et par le raisonnement probabiliste.

La notion de développement contingent va plus loin, car elle suppose que les effets imprévus peuvent dans certains cas être utilisés comme moteurs du développement. Une anecdote concernant le Khépéra mérite d'être racontée pour illustrer ces propos. Khépéra était utilisé avec un programme de phototaxie réflexe tout à fait élémentaire lui permettant de suivre une lampe torche. Un jour, alors que le porteur de la torche l'éteignit, le robot fit quelque chose de complètement inattendu qui marqua beaucoup tous les spectateurs. Le robot fit demi-tour sur place, se dirigea en évitant tous les obstacles vers une pièce éclairée, franchit la porte à moitié fermée de ce bureau en négociant parfaitement ce passage étroit pour se mettre à danser sous la lampe dont la lumière provenait, apparemment content d'avoir atteint son but ! Ce comportement étonnant est en fait facile à comprendre. La source de lumière principale étant le bureau éclairé, il est normal qu'elle ait attiré Khépéra. La combinaison du

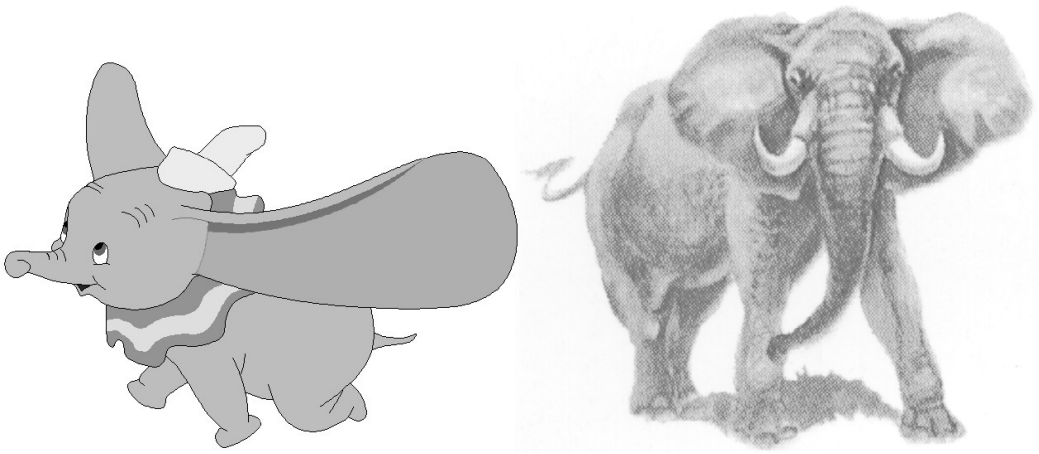


Figure 17 : les ailes n'ont pas été « conçues » pour que les animaux puissent voler : Disney n'a rien inventé

comportement de phototaxie et des ombres projetées par chaque obstacle explique que le robot en allant vers la lumière évite les obstacles et négocie facilement les passages étroits, le gradient de lumière lui indiquant toujours le chemin à suivre. Enfin, la « danse de victoire »

traduit seulement l'indécision du robot une fois arrivé dans le « marais » lumineux se trouvant sous la lampe. Cette expérience, déjà ancienne, fut à l'origine d'une bonne partie des travaux décrits ici, la question étant de savoir comment exploiter l'apparition de tels comportements émergents.

A.4.3 La remise en cause des connaissances préalables structurelles

Reprenant le schéma de l'anecdote qui vient d'être raconté, nous avons tout d'abord appris à Khépéra le comportement de phototaxie (en le pilotant avec le *joystick*). Nous obtenons ainsi une *description* P(VrotLum|D1C1) encodant les dépendances entre *Lum* et *Vrot* (voir figure 18.a).

Dans une deuxième phase, nous utilisons cette *description* de photaxie pour commander Khépéra (P(Vrot|LumD1C1)). Comme prévu, on constate que, placé dans un environnement avec une seule source de lumière assez rasante pour que les obstacles projettent des ombres importantes, Khépéra se dirige vers la lumière en évitant les obstacles. Alors que Khépéra exécute ce comportement, nous apprenons la *description* P(VrotDirProx|D2C2) encodant les dépendances entre *Vrot*, *Dir* et *Prox* (voir figure 18.b). Cette nouvelle *description* est utilisée pour commander Khépéra (P(Vrot|DirProxD2C2)). On vérifie qu'elle permet bien d'éviter les obstacles, et ce, même en l'absence d'ombre. On vérifie aussi que les valeurs de Vrot inférées à partir de la *description* de phototaxie P(VrotLum|D1C1) et à partir de la *description* d'évitement d'obstacle P(VrotDirProx|D2C2) sont les mêmes aux incertitudes près.

Enfin, dans une troisième phase (voir figure 18.c), on change la forme et la taille des ombres de certains obstacles en changeant leur hauteur. Par exemple, on prend des obstacles plus hauts qui projettent une ombre plus grande. Khépéra est commandé en mode « phototaxie » (P(Vrot|LumD1C1)). Les obstacles plus hauts ayant des ombres plus grandes, il les évite de plus loin que dans les deux premières expériences. Les valeurs de Vrot inférées à partir de la *description* de phototaxie P(VrotLum|D1C1) et à partir de la *description* d'évitement d'obstacle P(VrotDirProx|D2C2) ne sont donc plus du tout les mêmes. L'écart entre ces valeurs est représentatif de la hauteur des obstacles et peut éventuellement être utilisé pour construire une nouvelle*description* encodant les dépendances entre ces deux valeurs de *Vrot* et la hauteur des obstacles.

Les proximètres et les capteurs de lumières n'ont pas été « conçus » pour que Khépéra puisse mesurer la hauteur des obstacles. Ils ont été conçus pour des tâches tout autres. Pourtant cette expérience montre que, dans certaines conditions, ils permettent d'arriver à ce résultat.

L'ensemble des variables données au départ par le concepteur (connaissances préalables structurelles) peut s'avérer très rapidement insuffisant. Nous voyons dans l'expérience qui vient d'être décrite l'indication de la voie à suivre pour créer de nouvelles variables. Elle repose sur l'idée qu'une *description* une fois acquise peut être utile même dans des contextes différents de celui de l'apprentissage initial, en particulier, pour créer de nouvelles variables identifiant des « différences » entre la *description* de référence et les données en cours de traitement. Plus généralement, il nous semble ainsi possible d'envisager la création systématique et automatique de nouvelles variables potentiellement intéressantes.

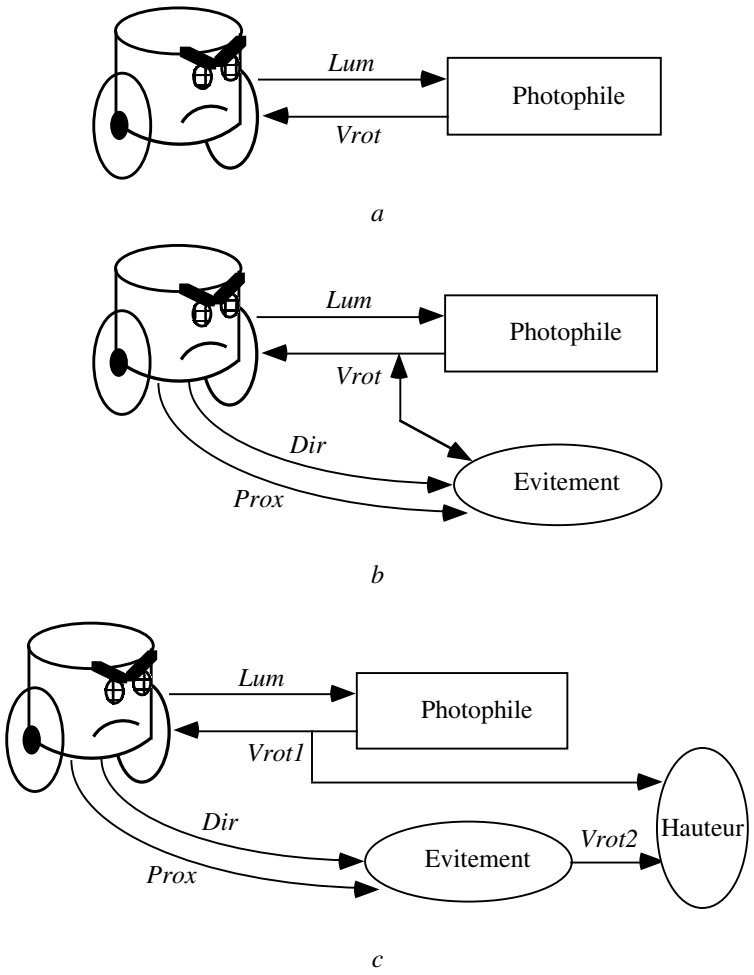


Figure 18 : mesurer la hauteur des obstacles avec des proximètres et des capteurs de lumières.

A.4.4 Vers une approche évolutionniste des connaissances préalables?

Un processus évolutionniste suppose 4 composantes fondamentales :
- Un code
- Un processus d'ontogenèse qui à partir du code soit capable par l'interaction avec l'environnement de produire des objets.
- Un processus d'échange d'information mutuelle qui à partir de 2 codes (ou plus) soit en mesure d'engendrer de nouveaux codes, assez ressemblant à leurs « ancêtres » pour en garder les acquis, mais assez différents d'eux pour explorer de nouvelles solutions.
- Un processus de sélection qui « évalue » ces objets et sélectionne les meilleurs en favorisant les codes correspondants dans le processus de génération de nouveaux codes.

Nous sommes avec l'approche F+D dans une situation qui rend apparemment possible la mise en place d'un processus évolutionniste pour le développement des facultés de perception, de décision et d'action d'un système sensori-moteur de type robotique.

Les connaissances préalables telles qu'elles ont été définies sont des objets mathématiques bien structurés. En tant que tels, il est certainement possible de les coder sous une forme convenant à un processus évolutionniste.

Par exemple, les connaissances préalables structurelles correspondent au choix d'un ensemble de variables parmi toutes les variables potentiellement utilisables par le système

sensori-moteur. Il existe de multiples manières connues de coder simplement le choix de n objets parmi m.

Les connaissances préalables de dépendances peuvent se traduire par le choix parmi les n variables précédemment sélectionnées d'un certain nombre de sous ensembles, eux-mêmes divisés en deux, entre les variables conditionnées et les variables conditionnantes. De multiples codes semblent, aussi ici, possibles.

Pour les connaissances a priori et les connaissances préalables d'observation, les formes de code possibles sont moins évidentes à imaginer, mais étant donné qu'il s'agit de connaissances formelles de type mathématique, trouver de tels codes est certainement possible.

L' inférence probabiliste et le principe de maximum d'entropie offrent le processus de développement du code des connaissances préalables par interaction avec les données expérimentales pour produire les objets appelés *descriptions*.

Dans la mesure où un code a été trouvé pour représenter un ensemble de connaissances préalables, il est facile d'imaginer divers opérateurs d'échanges d'information entre codes afin de produire de nouveaux codes correspondant à de nouveaux ensembles de connaissances préalables. Les travaux actuels sur les algorithmes génétiques proposent de nombreuses solutions qui devront être étudiées.

La partie la plus délicate à imaginer est sans aucun doute le processus de sélection des meilleures descriptions. Les critères de cette sélection ne sont pas encore forcément clairs et devront faire l'étude d'une étude approfondie. Cependant, plusieurs possibilités paraissent d'ores et déjà intéressantes à approfondir :

- Les *descriptions* « plates », les distributions de probabilités proches de la distribution uniforme, celles traduisant une très grande incertitude sont de manière évidente moins intéressantes que celles très « pointues » qui présentent des pics de probabilités. Les premières sont moins informationnelles que les secondes, et leur exploitation plus délicate et plus difficile. L'entropie semble, une fois de plus, une intéressante métrique pour quantifier cette notion, les distributions « plates » ayant une entropie beaucoup plus grande que les « pointues ». Les distributions très « plates » révèlent en général que les connaissances préalables ont été mal choisies (présence de variables cachées, mauvais choix de la structure de dépendance, etc...).
- Pour un jeu de données expérimentales, la proximité entre la *description* et les données, mesurée par ce qui est souvent appelé biais, semble souhaitable. Une méthode précise et rigoureuse pour quantifier ce deuxième critère est proposé dans l'article de ce même numéro « Interprétation ou Description (II) ».
- Inversement, avoir des *descriptions* complètement « irrégulières », sans aucune « continuité », ayant une très grande variance semble non souhaitable. Ces distributions sont en général le reflet exact des données d'apprentissage, mais offrent de très mauvaises capacités de « généralisation ».
- Enfin, pour pouvoir être exploitable en pratique les *descriptions* doivent être parcimonieuses dans l'emploi de la mémoire (représentation paramétrique économique) et permettre des calculs probabilistes faciles et surtout rapides. On retrouve ici la vieille notion de « rasoir d'Occam » qui propose de favoriser les représentations simples aux détriments des plus complexes.

Tous ces critères sont évidemment en contradiction les uns avec les autres, notamment le dilemme « biais versus variance » est bien connu et largement étudié (Geman92). Une large voie de recherche s'ouvre donc devant nous afin d'imaginer, étudier et implanter des processus de sélection intéressants. Un premier pas dans cette voie a été fait avec le DEA de Ludovic Mauconduit (Mauconduit97).

A.5 Bibliographie

Apostel Léo; *Syntaxe, sémantique et pragmatique* dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967a

Apostel Léo; *Logique et dialectique* dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967b

Balaniuk Remis; *Identification structurelle* ; Thèse de doctorat de l'INPG (Institut Natiaonal Polytechnique de Grenoble), Grenoble, France ; septembre 1996

Bessière Pierre; *Toward a synthetic cognitive paradigm: Probabilistic Inference ;* COGNITIVA 90, Madrid, 1990 ; edité par T. Kohonen & F. Fogelman-Soulie ; Elsevier science publishers, 1991

Bickhard Mark H. & Terveen Loren; *Foundational issues in artificial intelligence and Cognitive Science ; Impasse and solution* ; North-Holland, Elsevier ; 1995

Bradley B. & Swartz M. ; *Possible worlds ;* Oxford University Press, Great Britain, 1979

Braitenberg V.; *Vehicles: Experiments in synthetic psychology* ; MIT Press, Cambridge, USA, 1986

de Broglie Louis ; *Les représentations concrètes en micro-physique* ; dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967

Brooks, R.A. ; *Achieving artificial intelligence through building robots* ; AI MEMO 899, MIT, 1986a

Brooks, R.A. ; *A robust layered control system for a mobile robot*, IEEE journal of robotics and automation, RA-2, Avril 1986b

Brooks, R.A. ; *A robot that walks: emergent behaviors from a carefully evolved network* ; Neural Computations, 1:2, 1989

Brooks, R.A. ; *Elephants don't play chess*, in P. Maes (ed), *Designing autonomous agents: theory and practice from biology to engineeing and back,* MIT Press ; 1990

Brooks Rodney A.; *Artificial life and real robots* ; in ECAL 91 ;*Toward a practice of autonomous systems:* proc. of the 1st european conference on artificial life (Paris 1991), F. Varela & P. Bourgine (eds), MIT Press/Bradford books, 1991a

Brooks Rodney A.; *Intelligence without representation* ; Artificial Intelligence, n°47, 1991b

Brunel N., Nadal J.-P., and Toulouse G. ; *Information capacity of a perceptron* ; J. Phys. A: Math. and Gen., 25:5017--5037, 1992.

Campbell Jerremy ; *Gramatical Man* ; Simon & Schuster, 1982

Cheeseman P. ; *In defense of Probability* ; Proceedings of AAAI85, 1985

Cotrell Marie ; *Modèlisation des réseaux de neurones par des chaînes de Markov et autres application* ; Thèse d'état, Université Paris Sud (Orsay) ; 1988

Cox R. T. ; *The algebra of probable inference* ; The John Hopkins Press, Baltimore, 1961

Dedieu Eric ; *La représentation contingente : Vers une reconciliation des approches fonctionnelles et structurelles de la robotique autonome*. Thèse de troisième cycle INPG ; septembre 1995, Grenoble, France, 1995

de Finetti B. ; *Probability, induction and statistics* ; John Willey & sons, 1972

Dempster A.P. ; *A generalization of Bayesian inference* ; J. of the Royal Statistical Society, B 30 ; 1968

Dreyfus H. ; *What computers can't do : The limits of artificial intelligence* ; Harper and Row, New York, USA, 1979

Dubois D. & Prade H. ; *Fuzzy sets and systems: theory and applications* ; Academic Press, New York, 1980

Dubois D. et Prade H. ; *Théorie des possibilités* ; Masson ; Paris, France ; 1985

Ekeland Ivar ; *Le Calcul, l'Imprévu ; Les figures du temps de Kepler à Thom* ; Editions du Seuil, Paris, France ; 1984

Ekeland Ivar ; *Au hasard ; La chance, la science et le monde* ; Editions du Seuil, Paris, France ; 1991

Fodor Jeremy A. ; *The language of thought* ; Harvester Press ; Sussex ; 1976

Fodor Jerry A. ; *The Modularity of Mind - An essay on faculty psychology* ; A Bradford Book ; The MIT Press, Cambridge, Massachusetts, USA ; 1984

Fodor Jerry A. & Pylyshin Zenon W. ; *Connectionism and cognitive architecture : A critical analysis* in *Connections and Symbols* edited by S. Pinker & J. Mehler ; A Bradford Book, The MIT Press, Cambridge, Massachusetts, USA ; 1988

Foucault Michel ; *Les mots et les choses* ; Gallimard, Paris France ; 1966

Gardner E. ; *Optimal basins of attraction in randmly sparse neural network models ;* Journal of physics A, volume 22, number 12, 1989a

Gardner E. ; *Three unfinished works on the optimal storage capacity of networks ;* Journal of physics A, volume 22, number 12, 1989b

Geman S. & Geman D. ; *Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images ;* IEEE transactions on pattern analysis and machine intelligence, 6, 1984

Geman Stuart, Bienenstock Elie & Doursat René ; *NeuralNetworks and the Bias/Variance Dilemma* ; Neural Computation 4, 1-58 ; MIT Press ; 1992

Gibson J.J. ; *The ecological approach to visual perception* ; Houghton-Misslin company, Boston, USA, 1979
Goguen J.A. ; *L-fuzzy sets* ; Journal of mathematical analysis and applications, N°18, 1967
Gould Stephen J. ; *The flamingo's smile* ; edited by W.W. Norton, 1985 ; En français : *Le sourire du flamant rose* ; Editions du seuil, Paris ; 1988
Gould Stephen J. ; *Wonderful life* ; edited by W.W. Norton ; 1989
Gould Stephen J. ; *Bully for Brontosorus* ; Penguin editors ; 1991
Grassberger Peter & Nadal Jean-Pierre (editors) ; *From statistical physics to statistical inference and back* ; Kluwer Academic Publishers ; Dordrecht, The netherlands ; 1992
Grize Jean-Blaize ; *Historique. Logique des classes et des propositions. Logique des prédicats. Logiques modales* ; dans *Logique et connaissance scientifique* sous la direction de Jean Piaget ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967
Gutfreund H. & Toulouse G. (editors) ; *Biology and computation: a physicist's choice* ; Advances series in neuroscience, Vol. 3, World Scientific ; 1994
Harnad S. ; *Mind, Machines and Searle* ; Journal of theoretical and experimental artificial intelligence ; 1: 5-25 ; 1989
Harnad S. ; *The symbol grounding problem* ; Physica D, 42 : 335-346 ; 1990
Hervé T., Dolmazon J.M. & Demongeot J. ; *Random field and neural information ;* Proc. Natl. Acad. of Sciences, U.S.A., Vol 87, 1990
Hofstadter Douglas ; *Gödel, Escher, Bach : an Eternal Golden Braid* ; Basic Books Inc ; New York, U.S.A. ; 1979 ou *Gödel, Escher, Bach : les Brins d'une Guirlande Eternelle* ; InterEditions ; Paris, France ; 1985
Horswill Ian D. ; *Specialization of perceptual processes* ; MIT 1993
Jaynes E.T. ; *Probability theory - The logic of science* ; à paraître. Version provisoire disponible à http://bayes.wustl.edu. ; 1995
Kleene Stephen C. ; *Mathematical Logic* ; John Wiley & sons ; New York, USA ; 1967 ou *Logique mathématique* ; Armand Colin ; Paris, France ; 1971
Koza John ; *Evolving emergent wall following robotic behavior using the genetic programming paradigm* ; ECAL 91 ; Toward a practice of autonomous systems: proc. of the 1st european conference on artificial life (Paris 1991), F. Varela & P. Bourgine (eds), MIT Press/Bradford books, 1992
Labbi Abderrahim ; *Sur l'approximation et les systèmes dynamiques dans les réseaux neuronaux* ; Thèsede lInstitut National Polytechnique de Grenoble, Grenoble, France, 1993
Ladrière Jean ; *Les limites de la formalisation* ; dans *Logique et connaissance scientifique* sous la direction de Jean Piaget ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967
Lichnerowicz André ; *Remarques sur les mathématiques et la réalité* dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967
Lucas J.R. ; *Minds, Machines, and Godël* ; Philosophy 36 ; 1961
MacKay David J.C. ; *A practical bayesian framework for backprop networks* ; Neural computation ; 1992
Malcom C. & Smithers T.; *Symbol grounding via a hybrid architecture in an autonomous assembly system* ; DAI-RP 420, University of Edinburgh ; 1989
Mataric Maja J. ; *Navigating with a rat brain : a neurobiologically inspired model for robot spatial representation* ; SAB 90 From animals to animats: proc. of the 1st international conference on the simulation of adaptive behaviour (Paris 1990), J-A. Meyer & S. Wilson (eds), MIT Press/Bradford books, 1991
Mauconduit Ludovique; *Evolution de connaissances a priori : un pas vers l'autonomie* ; DEA INPG ; Grenoble ; 1997
McCarthy J. ; *Circumpscription: a form of non-monotonic reasoning* ; Artificial Intelligence n°13, 1980
McDermott D. & Doyle J. ; *Non-monotonic logic I* ; Artificial Intelligence, n°13, 1980
McDermott D. ; *Non-monotonic logic II* ; JACM n°29, 1982
Merleau-Ponty, M. ; *La structure du comportement. ;* 8ème ed., PUF, 1977
Merleau-Ponty M. ; *Phénoménologie de la perception* ; Editions Gallimard, Paris, France, 1945
Nadal Jean-Pierre ; *Réseaux de neurones : de la physique à la psychologie* ; Armand Colin, Paris, France ; 1992
Nadal J.-P. and Parga N. ; *Duality between learning machines: a bridge between supervised and unsupervised learning* ; LPSENS preprint, to appear in Neural Computation, 1993.
Nilsson Nils J. ; *Principles of Artificial Intelligence* ; Morgan Kaufmann Publishers ; U.S.A. ; 1980 ou *Principes d'intelligence artificielle* ; CEPADUES-EDITIONS ; Toulouse, France ; 1988
Omnès Roland ; *Philosphie de la science contemporaine* ; Gallimard, Paris, France ; 1994
Pearl Judea ; *Probabilistic reasoning in intelligent systems : Networks of plausible inference* ; Morgan Kaufmann Publishers, San Mateo, California, USA ; 1991
Piaget Jean (sous la direction de) ; *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967a
Piaget Jea ; *Les problèmes principaux de l'épistémologie des mathématiques* dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967b
Piaget Jean ; *Les données génétiques de l'épistémologie physique* ; dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967c

Piaget Jean ; *Les relations entre le sujet et l'objet dans la connaissance physique* ; dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967d

Poincaré Henri ; *La science et l'hypothèse* ; Publié originalement en 1902 ce livre est maintenant réédité aux éditions de la Bohème, Rueil-Malmaison, France ; 1992

Proust Joëlle ; *Questions de forme : Logique et proposition analytique de Kant à Carnap* ; Fayard, 1986

Reeke G. N. & Edelman G. M. ; *Real brains and artificial intelligence* in *The AI debate : false starts, real fundations edited by* S.R. Graudebard ; The MIT Press, Cambridge, USA ; 1988

Richalet, J. 1991 *Pratique de l'identification.* ADERSA, coll. Traité des nouvelles technologies, Hermès.

Robert Claudine ; *Observable networks, Applications to medical decision making* in Proceedings of IEEE PAMI, 1990a

Robert Claudine ; *An entropy concentration theorem: applications in artificial intelligence and descriptive statistics* ; Journal of Applied Probabilities ; september 1990b

Robert Claudine ; *Modèles statistiques pour l'intelligence artificielle : l'exemple du diagnostic médical* ; Masson, Paris, France ; 1991

Rosser J.B. & Turquette A.R. ; *Many-valued logics* ; North-Holland, Amsterdam, Nederlands, 1958

Searle J. R. ; *Mids, Brains, and Programs* in *Mind design* edited by J. Haugeland ; MIT Press ; 1981

Shafer G. ; *A mathematical theory of evidence* ; Princeton University Press, Princeton, USA, 1976

Smolensky P. ; *Information processing in dynamical systems: foundations of Harmony theory* in *Parallel Distributed Processing* by D.E. Rumelhart, J.L. McClelland & the P.D.P. Research Group ; The M.I.T. Press, 1986

Stewart J. ; *Epistemologie des sciences de la cognition et du langage* ; Annales Carnac, 5, 115-134 ; Lausanne, Suisse, 1992

Suchman Lucy A. ; *Plans and situated actions ; The problem of human/machine communication* ; Cambridge University Press, Great Britain, 1987

Tarentola Albert ; *Inverse Problem Theory ; Methods for data fitting and model parameters estimation* ; Elsevier ; New York, USA ; 1987

Turner Raymond ; *Logics for artificial intelligence* ; Ellis Horwood limited, New York, USA, 1984

Ulmo Jean ; *Les concepts physiques* ; dans *Logique et connaissance scientifique* ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967

Varela Francisco ; *Connaître les sciences cognitives ; tendances et perspectives* ; Collection « Science ouverte », Editions du Seuil, Paris, France, 1989a

Varela Francisco ; *Autonomie et Connaissance ; Essai sur le vivant* ; Editions du Seuil, Paris, France, 1989b

Varela Francisco ; ECAL 91 *Toward a practice of autonomous systems: proc. of the 1st european conference on artificial life (Paris 1991),* F. Varela & P. Bourgine (eds), MIT Press/Bradford books, 1991

Varela Francisco, Thompson Evan & Rosch Eleanor ; *L'inscription corporelle de l'esprit ; science cognitives et expérience humaine* ; Editions du Seuil, Paris, France, 1993

Vauquois Bernard ; *Calculabilité des langages* ; Université de Grenoble ; Grenoble, France ; 1976

Winograd T. & Flores F. ; *Understanding computers and cognition : A new foundation for design* ; Ablex Press, USA, 1986

Zadeh L. A. ; *Fuzzy sets* ; Information and Control, N°8, 1965

Zadeh L. A. ; *Fuzzy sets as a basis for a theory of possibility* ; Fuzzy Sets and Systems, N°1, 1978

Annexe B

Cette annexe présente l'article « *Interprétation ou Description (II) : Fondements mathématiques de l'approche F+D* » parue, en 1998, dans Intellectica, revue de l'Association pour la Recherche Cognitive :

P. Bessière, E. Dedieu, O. Lebeltel, E. Mazer & K. Mekhnacha ; *Interprétation ou Description (II) : Fondements mathématiques de l'approche F+D* ; Intellectica, Revue de l'Association pour la Recherche Cognitive ; 1998/1-2, 26-27, pp. 313-336 ; 1998

Pierre Bessière
Eric Dedieu *Laboratoire LEIBNIZ - CNRS*
Olivier Lebeltel
Emmanuel Mazer
Kamel Mekhnacha

INTERPRÉTATION OU DESCRIPTION (II)

Fondements mathématiques de l'approche F+D

RÉSUMÉ : Cet article fait suite à l'article « Interprétation ou Description (I) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs ». Son objectif est de présenter les fondements mathématiques de l'approche F+D présenter dans le premier article. Les deux composantes fondamentales dont nous avons besoin pour notre approche F+D sont présentées. D'une part, des règles formelles permettant de raisonner sur des données incertaines et incomplètes. D'autre part, le principe de maximum d'entropie, qui permet de clarifier le lien entre descriptions et expériences et donne un cadre théorique général pour l'apprentissage.

MOTS-CLÉS : science cognitive, intelligence artificielle, robotique, logique, inférence probabiliste, inférence bayésienne, entropie maximum, perception, action, décision

ABSTRACT : This paper is a continuation of the paper entitled « Interprétation ou Description (I) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs ». Its purpose is to introduce the mathematical basis of the F+D approach presented in the first paper. The two fundamental components needed in our approach are discussed. On the one hand, the formal rules to reason about uncertain and incomplete knowledge. On the other hand, the maximum entropy principle, which clarifies the link between "descriptions" and experiments and offers a clear and sound theoretical framework for learning.

KEY WORDS : cognitive science, artificial intelligence, robotics, logic, probabilistic inference, bayesian inference, maximum entropy, perception, action, decision

B.1 PaL : un fondement mathématique pour l'approche F+D

La théorie des probabilités n'est rien d'autre que le sens commun fait calcul.

Laplace

The actual science of logic is conversant at present only with things either certain, impossible, or entirely doubtful, none of which (fortunately) we have to reason on. Therefore the true logic for this world is the calculus of Probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man's mind[26].

James Clerk Maxwell[27]

By inference we mean simply: deductive reasoning whenever enough information is at hand to permit it; inductive or probabilistic reasoning when - as is almost invariably the case in real problems - all the necessary information is not available. Thus the topic of « Probability as Logic » is the optimal processing of uncertain and incomplete knowledge[28].

E.T. Jaynes

Cet article fait suite et complète l'article intitulé « *Interprétation ou Desciption (I) : proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs* » paru dans ce même numéro. L'objectif poursuivi est de présenter les principes des fondements mathématiques de l'approche F+D.

E. T. Jaynes propose dans (Jaynes95) une théorie de la cognition, fondée sur les probabilités, appelée « Probability as Logic » (*PAL*).

---

[26] La science logique contemporaine parle uniquement de faits certains, impossibles, ou complètement douteux - sujets sur lesquels nous n'avons (heureusement) jamais à raisonner. En conséquence, la vraie logique pour notre monde est le calcul des probabilités, qui prend en compte une quantification des plausibilités qui sont, ou devraient être, dans tout esprit rationnel.

[27] Citation empruntée à (Jaynes95)

[28] Par inférence, nous entendons le raisonnement déductif quand on dispose d'assez d'information pour le permettre; le raisonnement inductif ou probabiliste quand, comme c'est presque toujours le cas pour les problèmes réels, toute l'information nécessaire n'est pas disponible. C'est pourquoi le propos de « Probability as Logic » est le traitement optimal de l'information incertaine et incomplète.

Cette théorie fournit les deux composantes fondamentales dont nous avons besoin pour notre approche F+D, d'une part, des règles formelles permettant de raisonner sur des données incertaines et incomplètes, d'autre part, le principe de maximum d'entropie, qui permet de clarifier le lien entre *descriptions* et expériences et donne un cadre théorique général pour l'apprentissage.

B.2 Raisonner malgré l'incertitude et l'incomplétude : l'inférence probabiliste.

Le raisonnement probabiliste peut-être fait par un système de calcul formel. Il est fondé sur les probabilités alors que la plupart des systèmes formels sont basés sur la logique. Le raisonnement probabiliste permet de calculer pour chaque proposition une valeur réelle comprise entre 0 et 1, interprétable comme une plausibilité. Les systèmes formels courants servent, eux, à dériver ou réfuter les propositions, ce qui revient à calculer une valeur qui est soit 0, soit 1.

B.2.1 Le théorème de Cox

Le théorème de Cox est le résultat fondamental montrant comment la notion intuitive de plausibilité se formalise par la notion mathématique de probabilité.

Notons $\pi(A|C)$ la plausibilité d'une proposition A, jugée au vu d'un ensemble de connaissances C. Les postulats[29] suivants explicitent cette notion de plausibilité et permettent de dériver la forme mathématique exacte qu'elle doit prendre :

- Les plausibilités sont représentées par des nombres réels (i.e. $\pi(A|C)$ est un réel)[30].
- S'il existe plusieurs calculs corrects pour déterminer une plausibilité, ils doivent tous mener au même résultat (propriété dite de « consistance »).
- Si de nouvelles informations C' viennent remplacer les informations C de façon à augmenter la plausibilité de A, alors la plausibilité de la proposition contraire ¬A doit diminuer. Formellement :

$$\text{si } \pi(A|C') > \pi(A|C), \text{ alors } \pi(\neg A|C') < \pi(\neg A|C) \qquad [4.1]$$

- Si de nouvelles informations augmentent la plausibilité de A mais ne concernent en rien une autre proposition B, alors la plausibilité de la conjonction de A et B (noté AB) ne peut qu'augmenter. Formellement :

$$\text{si } \begin{cases} \pi(A|C') > \pi(A|C) \\ \pi(B|AC') = \pi(B|AC) \end{cases} \text{, alors } \pi(AB|C') \geq \pi(AB|C) \qquad [4.2]$$

A partir de ces postulats de base R.T. Cox a montré que le raisonnement plausible devait obligatoirement suivre 2 règles à partir desquelles toute la théorie des probabilités peut être reconstruite.

La règle qui donne la probabilité d'une conjonction :

$$P(AB|C) = P(A|C) \times P(B|AC) = P(B|C) \times P(A|BC) \qquad [R1]$$

---

[29] Exprimés d'après (Jaynes95)

[30] Par commodité, nous représenterons une plus grande plausibilité par un nombre plus grand — mais ce choix est une pure convention.

Et la règle qui exprime que la somme des probabilités d'une proposition et de sa négation est égale à 1 :

$$P(A|C) + P(\neg A|C) = 1 \qquad [R2]$$

Ce théorème montre de plus que toute technique de calcul de plausibilité qui ne vérifierait pas ces deux règles, enfreindrait nécessairement l'un au moins des postulats posés au paragraphe précédent. Donc si l'on accepte ces postulats, le calcul des probabilités, et lui seul, permet de faire du raisonnement plausible.[31]

On déduit très aisément des règles R1 et R2 la règle qui donne la probabilité de la disjonction (A+B):

$$P(A + B|C) = P(A|C) + P(B|C) - P(AB|C) \qquad [R3]$$

B.2.2 Les syllogismes du raisonnement probabiliste

La déduction logique est essentiellement fondée sur les deux règles suivantes :
- Modus ponens : si « A est vrai » et si « Si A est vrai alors B est vrai » alors « B est vrai ».
- Modus tollens : si « B est faux » et si « Si A est vrai alors B est vrai » alors « A est faux ».

Cependant, nous employons constamment des critères plus « faibles » que ces deux règles :

Le premier peut être énoncé de la manière suivante : si « Si A est vrai alors B est vrai » et si « B est vrai » alors « A devient plus plausible ». La vérification que la conséquence d'une implication est vraie renforce la confiance que l'on a dans la prémisse de cette implication.

Un exemple de ceci est donné par les propositions suivantes :
- A ≡ « La valeur du proximètre avant gauche du Khépéra (V3) est supérieure à 10 » ;
- B ≡ « Il y a un obstacle proche droit devant ».

La présence de l'obstacle (B vrai) ne garantit pas totalement que le capteur V3 réponde fortement[32] (A vrai), mais contribue certainement à nous convaincre que la valeur de V3 doit être élevée (A devient plus plausible).

Le second critère faible s'énonce : si « Si A est vrai alors B est vrai » et si « A est faux » alors « B devient moins plausible ». Supprimer une des prémisses qui pouvaient impliquer la vérité d'un fait donné fait diminuer notre confiance dans cette vérité. Dans l'exemple précédent, si le proximètre V3 ne répond pas, nous aurons plus de mal à nous persuader qu'il peut y avoir un obstacle.

Enfin, le troisième critère est plus faible encore : si « Si A est vrai alors B devient plus plausible » et si « B est vrai » alors « A devient plus plausible ». C'est typiquement celui qui sous-tend les enquêtes policières et les décisions de justice. « Si X est coupable alors il est probable que X ait laissé ses empreintes sur l'arme du crime », « X a laissé ses empreintes sur l'arme du crime » donc il devient plus plausible que « X soit coupable ». En fait, l'exemple pris pour illustrer les 2

---

[31] Une autre manière de voir ce résultat, consiste à se demander, pour chaque théorie non probabiliste des plausibilités, lequel des 4 postulats n'est pas verifié. Une fois répondu à cette question, on peut discuter de manière plus saine et plus sereine sur les inconvénients et avantages relatifs de la dite théorie et du calcul des probabilités.

[32] Par exemple, le capteur ne répond pas aux obstacles trop sombres absorbant les infrarouges.

règles précédentes est plutôt de cette nature, car la réponse élevée du proximètre avant gauche n'implique pas de manière absolue la présence d'un obstacle. Des exceptions à une telle implication sont toujours facilement imaginables[33].

La logique formelle est incapable de prendre en compte ces critères « faibles » pourtant si utiles en pratique. Le raisonnement probabiliste les intègre naturellement. De plus, et c'est très important, il apparaît que si on se limite à des propositions de probabilité 0 ou 1 (certainement vraies ou certainement fausses), les règles du raisonnement probabiliste permettent de tenir tout raisonnement logique souhaité - et c'est pourquoi l'approche F+I est un cas particulier de l'approche F+D.

B.2.3 Problèmes directs et inverses

Pour une *description* donnée, on peut distinguer les deux ensembles de variables suivants :
- d'une part, un ensemble Δ de variables « observables » dont les valeurs peuvent être « mesurées » par un quelconque moyen ;
- d'autre part, un ensemble Γ de variables non directement accessibles, dont les valeurs peuvent uniquement être « estimées » par le raisonnement ou données *a priori*.

Soit D un ensemble de mesures des variables de Δ appelé ensemble de données, soit C un ensemble de valeurs pour les variables de Γ appelé ensembles de paramètres ou connaissances préalables, on s'intéresse à l'étude des probabilités conjointes de D et C.

La règle [R1] nous donne :

$$P(DC) = P(C) \times P(D \mid C) = P(D) \times P(C \mid D)$$

Si l'on admet connaître les valeurs de certains paramètres et si l'on fixe par hypothèse celles des autres, le « problème direct » consiste à « prédire » les valeurs des données. Mathématiquement, cela signifie que l'on cherche $P(D \mid C)$.

Inversement, si l'on connaît un ensemble de données et que l'on cherche $P(C \mid D)$ on traite le « problème inverse ». Il consiste à chercher les paramètres qui rendent « au mieux » compte des expériences ou encore à choisir parmi plusieurs hypothèses envisageables laquelle est la plus probable. On voit qu'en particulier, le problème inverse recouvre les problèmes de type identification dont il a été question au paragraphe précédent.

Pour illustrer le problème direct et le problème inverse et pour donner quelques exemples élémentaires de raisonnement probabiliste, plaçons-nous dans le cas ou notre système est une urne de Bernouilli remplie de boules blanches et noires dans laquelle on effectue des tirages sans remise.

La connaissance préalable C s'exprime par les énnoncés N ≡ « L'urne contient n boules » et par B ≡ « L'urne contient b boules blanches », paramétrées par n et b.

Les données D peuvent s'exprimer, par exemple, sous la forme $T_i \equiv$ « On observe le tirage d'une boule blanche au $i^{ème}$ tirage ».

Le problème direct s'intéresse aux probabilités de D, c'est-à-dire de T_i, sachant C, c'est-à-dire n et b.

---

[33] Par exemple, le capteur V3 est « grillé » et indique la valeur maximum de 15 en toute circonstance.

Considérons tout d'abord à la probabilité $P(T_2 \mid C)$ de tirer une boule blanche au deuxième tirage. Nous pouvons écrire que $T_2 = (T_1 + \neg T_1)T_2 = T_1T_2 + \neg T_1T_2$. En appliquant la règle [R3] puis la règle [R1] on obtient :

$$\begin{aligned} P(T_2 \mid C) &= P(T_1T_2 \mid C) + P(\neg T_1T_2 \mid C) \\ &= P(T_2 \mid T_1C)P(T_1 \mid C) + P(T_2 \mid \neg T_1C)P(\neg T_1 \mid C) \\ &= \left(\frac{b-1}{n-1} \times \frac{b}{n}\right) + \left(\frac{b}{n-1} \times \frac{n-b}{n}\right) \\ &= \frac{b}{n} \\ &= P(T_1 \mid C) \end{aligned} \qquad [4.3]$$

On vérifie bien que la probabilité d'avoir une boule blanche au deuxième tirage (ne sachant pas le résultat du premier) est la même que celle d'avoir une boule blanche au premier tirage. Il en est d'ailleurs de même pour tous les tirages.

Intéressons-nous maintenant à $P(T_1 \mid T_2C)$ la probabilité de tirer une boule blanche au premier tirage sachant le tirage d'une boule blanche au deuxième.

La règle [R1] nous donne :

$$P(T_1T_2 \mid C) = P(T_1 \mid C)P(T_2 \mid T_1C) = P(T_2 \mid C)P(T_1 \mid T_2C) \qquad [4.4]$$

Comme on vient de démontrer en [4.3] que :

$$P(T_1 \mid C) = P(T_2 \mid C)$$

On obtient:

$$P(T_1 \mid T_2C) = P(T_2 \mid T_1C) \qquad [4.5]$$

Ce raisonnement peut se reproduire pour un i et un j quelconque et on obtient finalement :

$$P(T_i \mid T_jC) = P(T_j \mid T_iC) \qquad \forall i, \mathrm{j} \qquad [4.6]$$

Ce résultat peut surprendre, essentiellement parce qu'il peut paraître surprenant que la probabilité du tirage d'une boule blanche au premier tirage puisse être influencée par le deuxième tirage. Une idée très profondément ancrée dans nos esprits cartésiens et newtoniens est qu'un événement ne peut être influencé que par les événements qui l'ont précédé dans le temps. En fait, contrairement aux apparences, ce résultat ne remet pas en cause ce principe de la physique. La confusion vient de ce que l'on considère que le premier tirage a été influencé par le second alors que ce que nous dit le résultat [4.5] c'est que la connaissance que l'on a de ce qui peut se passer au premier tirage est éventuellement influencée par la connaissance du résultat du deuxième tirage. Il faut bien distinguer entre la causalité physique qui ne peut que se propager vers le futur et la capacité d'inférence qui fonctionne aussi bien dans un sens que dans l'autre.

Le calcul qui a conduit à [4.5] est particulièrement simple et résulte d'un usage tout à fait élémentaire des règles des probabilités. Pourtant, même ce résultat trivial n'est compréhensible que si l'on considère une distribution de probabilité comme l'état de connaissance d'un individu sur un certain phénomène et non pas comme une *description* de ce phénomène indépendamment de

l'observateur. Ce seul exemple pose déjà un sérieux problème pour toute épistémologie objectiviste des probabilités.

Plaçons-nous maintenant, pour illustrer le problème inverse, dans le cas où n le nombre total de boules est connu mais où b le nombre de boules blanches n'est pas connu. On va chercher à cerner b grâce à un ensemble de données D constitué avec des tirages issus de cette urne.

Alors que pour le problème direct, nous nous intéressions à la probabilité de D sachant C=NB, nous nous intéressons, maintenant à la probabilité de B sachant D et N, d'où le nom de problème inverse.

La règle [R1] nous donne :

$$P(BD \mid N) = P(B \mid N) \times P(D \mid BN) = P(D \mid N) \times P(B \mid DN) \qquad [4.7]$$

d'où on tire :

$$P(B \mid DN) = P(B \mid N)\frac{P(D \mid BN)}{P(D \mid N)} \qquad [4.8]$$

On constate que le résultat dépend à la fois de P(D | BN) probabilité directe, et de P(B | N)[34] : cette distribution, dite « *a priori* » depuis la fin du XVIII^{e} siècle, doit être connue - donnée ou calculable - pour qu'on puisse effectuer le calcul. Quant à P(D | N), elle peut toujours être obtenue par normalisation en sommant P(B | DN) sur toutes les valeurs de B possibles. Ainsi le résultat d'une série d'expériences (D) n'est pas interprétable en soi, mais uniquement au jour de connaissances préalables apportées ici par la distribution *a priori*. Voici un deuxième exemple de ce qui vient juste d'être dit, à savoir que le raisonnement probabiliste ne peut pas être la description d'une quelconque réalité indépendante de l'observateur, mais nécessite pour interpréter toute donnée la présence d'informations préalables sur l'état de connaissance de cet observateur.

Il serait trop long de développer ici les calculs pour les différentes distributions *a priori* envisageables. Nous invitons donc le lecteur à se reporter au chapitre 6 de (Jaynes95) pour satisfaire sa curiosité concernant P(B|DN).

B.2.4 Moteur d'inférence probabiliste

Un moteur d'inférence et d'apprentissage bayésien a été développé. Ce moteur a pour ambition d'être capable d'automatiser les raisonnements probabilistes. Il peut être, en cela, comparé aux nombreux moteurs d'inférences logiques qui existent et permettent d'automatiser le raisonnement symbolique (voir, par exemple, Prolog (Colmerauer86), LogLisp (Robinson83a) ou Lolita (Bessière87)).

Une première version du moteur d'inférence a été développée, fonctionne et est utilisée. Son principe, très simplifié, est le suivant :

Si l'on connait :

---

[34] Distribution de probabilité traduisant la connaissance a priori que l'on a du nombre b de boules blanches dans l'urne sachant le nombre n total de boules.

- $P(X_1, X_2 ..., X_n)$ la distribution de probabilité conjointe sur les variables X1, X2, ... Xn sous la forme d'une décomposition de cette distribution conjointe en un produit de distributions plus simples, par exemple $P(X_1, X_2 ..., X_n) = P(X_1)P(X_2 | X_1)P(X_3 | X_2)...P(X_n | X_{n-1}X_2)$;
- et, les formes paramétriques identifiées de toutes les distributions de ce produit ;

le moteur d'inférence est alors capable de répondre à toute question concernant la distribution conjointe.

Par toute question, on entend n'importe quelle expression de la forme $P(X_i ... X_j | X_k ... X_l)$ avec un nombre arbitraire de variables à gauche, un nombre arbitraire de variables à droite et un nombre arbitraire de variables inconnues (absentes).

La solution générale a une telle question est donnée par :

$$P(X_i ... X_j | X_k ... X_l) = \frac{1}{Z} \sum_{\text{variables inconnues}} P(X_1)P(X_2 | X_1)P(X_3 | X_2)...P(X_n | X_{n-1}X_2)$$

avec Z une constante de normalisation.

Pour une question donnée, cette expression est tout d'abord simplifiée symboliquement par le moteur. Sa valeur numérique est ensuite estimée. La complexité du calcul numérique vient évidemment des sommes. Nous avons développé dans le moteur une méthode d'estimation de ces sommes très performante qui fait notamment appel à un algorithme d'optimisation de type « génétique » et à une représentation des distributions sous forme très compacte et efficace, inspirée des Octree.

Il est essentiel de remarquer que dans l'approche préconisée ici toutes les variables jouent des rôles mathématiques parfaitement équivalents. Le moteur d'inférence étant capable de répondre à n'importe quelle question, il n'y a pas à proprement parler de problèmes directs et de problèmes inverses. Ceci est sans doute l'une des qualités essentielles de l'inférence probabiliste.

B.3 Ancrer les descriptions : le principe de maximum d'entropie

La deuxième composante nécessaire à notre approche F+D est de pouvoir disposer d'un moyen générique de construire les descriptions.

Étant donné un ensemble de connaissances préalables C et de données expérimentales D, il existe en général de très nombreuses (voir une infinité) de distributions de probabilités (descriptions) compatibles à la fois avec ces connaissances et ces données[35]. Toutefois, toutes ces descriptions ne sont pas équivalentes. Certaines semblent plus « cohérentes », plus « probables », plus « intéressantes » que d'autres. Cet « intérêt » peut être jugé dans un sens mathématique précis par la fonction entropie H. Pour une distribution discrète P qui assigne, à q propositions, les probabilités $\{p_1, ..., p_q\}$, l'entropie est définie par :

$$H(P) = -\sum_{i=1}^{q} p_i \log(p_i) \qquad [4.11]$$

Le principe de maximum d'entropie affirme que parmi les distributions compatibles avec un ensemble de connaissances préalables C et de données expérimentales D, la meilleure possible est

[35] Par exemple, il existe une infinité de distributions ayant même moyenne et même écart-type. La gaussienne est toujours choisie dans ce cas car c'est justement la distribution qui maximise l'entropie de cette classe.

celle qui maximise l'entropie H. Ce principe donne donc le moyen théorique et générique recherché pour construire les descriptions. Avant d'en justifier l'emploi présentons d'abord un exemple très simple afin de bien comprendre à quoi il sert et comment il peut être utilisé.

B.3.1 Vingt mille chiffres, pour quoi faire?

Supposons que nous ayons à notre disposition un ensemble de données expérimentales brutes sous la forme de 20 000 chiffres (voir figure 16). Un tel jeu de données est manifestement inexploitable tel quel. Sans connaissances préalables, ce n'est qu'une suite de signes cabalistiques sans aucune espèce de signification possible.

Une connaissance préalable minimale absolument requise consiste à préciser comment ces 20 000 chiffres se répartissent pour former des nombres. Supposons qu'ici chaque chiffre soit un nombre : on a donc 20 000 nombres correspondant à 20 000 valeurs d'un certain nombre, disons n, de variables.

Une deuxième connaissance préalable nécessaire est de garantir que ces 20 000 données forment un tout cohérent, correspondant bien à l'observation d'un seul et même phénomène et qu'il convient d'essayer de les analyser ensemble. [36]

Une troisième connaissance préalable consiste à préciser le nombre n des variables et à répartir entre elles les 20 000 valeurs. Supposons qu'ici une seule variable V ait été retenue. Nous avons donc 20 000 expériences concernant V.

Enfin, une quatrième connaissance préalable tout à fait déterminante suppose que l'ordre (temporel et/ou spatial) dans lequel ces 20 000 expériences se présentent n'est pas pertinent.[37]

Par ces choix, nous avons défini l'espace des phases et placé les points expérimentaux. Les connaissances préalables de ce type ont été qualifiées de structurelles dans l'article « *Interprétation ou Description* (I) ».

---

[36] Cette exigence peut paraître évidente et triviale. Pourtant les physiciens et les biologistes savent bien l'extrême difficulté qu'il peut y avoir à garantir l'intégrité d'un jeu de données vis-à-vis d'un phénomène observé.

Le choix des données à traiter ensemble est, en fait, fondamental car il définit implicitement le phénomène qu'on veut étudier.

Si, par exemple, on sépare le jeu de 20 000 données en deux parties, cela signifie qu'on suppose qu'une variable « cachée » (non mesurée) a changé d'état et qu'on préfère étudier ce qu'on a observé comme 2 phénomènes séparés.

C'est aussi souvent à ce stade que certaines données sont écartées car jugées « aberrantes ». En rejetant ces données comme « extérieures » au phénomène, on définit subjectivement ses limites.

[37] Ce choix est très « compromettant », car il suppose que les dépendances qui existent entre ces 20 000 valeurs ne sont d'origine ni spatiale ni temporelle.

6344034536465615115606153406464146525152103362300504630032225441211305230131115445034143031035463116352614542100631153542210661106316651636113450363146622135544356655056156633461023164002653513524011554420061634142341621514660540561143041210200406211043400243650012335630533252066442434403653656211530025634121105252061250043353230321141066430655314144236213133323264066442463045532404213335535105164041113611540146215543115111360013443564426154542565165115422605365451110625366344244502345260066161026444222522250325100603032516035231635603116035231636503116502443042452033211465431105533505414524266356451110625366344244502345260066161026444222522250325100603032516

(et donc pas une description) mais il permet une réduction, éventuellement considérable, de l'information.

Pour pouvoir produire notre première description, il faut encore faire une hypothèse supplémentaire, à savoir le nombre de valeurs possibles que peut prendre la variable observée. On peut alors démontrer (Jaynes95) que la meilleure description possible est la loi de succession de Laplace :

$$P([V=i]) = \frac{1+n_i}{\langle\Omega\rangle + n} \qquad [4.12]$$

avec <Ω> le nombre de valeurs possibles pour V, n_i le nombre d'expériences où la valeur i a été mesurée et n le nombre d'expériences totales.

Cette loi de succession de Laplace se rapproche de l'histogramme dès que le nombre d'expériences n devient grand devant le nombre de cas possible <Ω>. Elle présente, cependant, deux avantages sur l'histogramme : elle se comporte bien quand le nombre d'expériences n est faible (notamment, on retrouve la distribution uniforme pour $n=n_i=0$) et, surtout, elle ne fixe la probabilité d'aucun événement à 0, évitant par là un choix définitif devant reposer sur une certitude (une probabilité de 0 ne peut pas évoluer par apprentissage et rend impossible tout événement dépendant de l'événement de probabilité nulle). Cette loi de succession de Laplace est la forme la plus rudimentaire de description de données expérimentales. C'est elle qui suppose le moins de connaissances préalables et, donc, « colle » au plus près des données. C'est elle, aussi qui nécessite le plus de paramètres étant donné que pour coder n données il faut <Ω> paramètres.

Supposons que nous apprenions maintenant que la variable V peut prendre une quelconque des 6 valeurs entre 1 et 6 (<Ω>=6). Nous allons alors pouvoir calculer effectivement les probabilités pour chaque valeur de V (voir ligne L du tableau 1). Notre collection de données expérimentales commence à « prendre du sens » et à permettre le raisonnement. Il devient notamment possible de faire des prédictions sur les résultats à venir.

La loi de succession de Laplace s'avère rapidement inadaptée dès que l'on a des connaissances préalables plus riches. De ce fait, elle a souvent été critiquée dans l'histoire des probabilités. Par exemple, la loi de succession de Laplace attribue une probabilité de 11/12 à un enfant de dix ans de vivre une année de plus (<Ω>=2, n_i=10, n=10) alors qu'elle attribue pour son grand-père de 70 ans une probabilité de 71/72 au même événement (<Ω>=2, n_i=70, n=70). Ce résultat nous choque par son inexactitude évidente. Cependant, ce n'est pas la loi de succession de Laplace qui est en cause : c'est simplement que pour juger de la validité de ce résultat, nous disposons d'informations préalables qui n'ont pas été prises en compte pour établir ces probabilités.

Curieusement, face à des problèmes de ce genre, de nombreux auteurs ont jugé que les calculs utilisant la loi de succession de Laplace devaient être faux, ou ont invoqué d'obscurs principes pour bannir l'utilisation de cette distribution, alors qu'il leur suffisait de constater que son emploi n'était tout simplement pas en accord avec leurs prémisses.

Avant de pouvoir pousser notre analyse des 20000 données en instillant petit à petit de nouvelles connaissances préalables, il faut présenter quelques points techniques supplémentaires concernant le maximum d'entropie. Mathématiquement, rechercher la distribution maximisant l'entropie H revient à résoudre un problème d'optimisation sous contraintes. Bien qu'en théorie le principe de maximum d'entropie soit applicable à n'importe quel type de connaissances préalables, il s'avère qu'en pratique, la plupart du temps, on ne lui connaît pas de solution analytique.

Il existe cependant un type de contraintes (connaissances préalables) très souvent rencontré dans la pratique (notamment en physique) pour lequel on connaît la forme analytique de la distribution qui correspond au maximum d'entropie. Il s'agit du cas où les connaissances préalables C sont données sous forme de m fonctions réelles f_j, nommées *observables*, et dont on peut calculer les moyennes F_j sur les données expérimentales. On assume alors que les *descriptions* adéquates du phénomène sont données par les distributions P pour lesquelles les espérances des fonctions f_j sont égales aux moyennes expérimentales F_j.

$$\forall j, j \in \{1,\ldots,m\};\ \sum_{v \in D} P(v) f_j(v) = F_j \qquad [4.13]$$

La distribution de maximum d'entropie correspondant à ces contraintes est alors P* donnée par :

$$P^*(v) = \frac{1}{Z(\lambda_1,\ldots,\lambda_m)} e^{-\sum_{i=1}^{m} \lambda_i f_i(v)} \qquad [4.14]$$

où Z désigne la fonction de partition, donnée par :

$$Z(\lambda_1,\ldots,\lambda_m) = \sum_{v \in \Omega} e^{-\lambda_1 f_1(v) - \ldots - \lambda_m f_m(v)} \qquad [4.15]$$

les m coefficients λ_i pouvant être déterminés par les m équations traduisant les contraintes [4.13]:

$$\frac{\partial}{\partial \lambda_j} ln\left(Z\left(\lambda_1,\ldots,\lambda_j,\ldots,\lambda m\right)\right) + F_j = 0 \qquad [4.16]$$

Ce type de distributions « exponentielles » permet de coder n données avec m paramètres qui sont, au choix, ou bien les F_i appelés « niveaux de contraintes » des observables, ou bien les λ_i appelés « multiplicateurs de Lagrange ».

Il existe en physique un très grand nombre de telles distributions de maximum d'entropie correspondant au choix de tel ou tel ensemble d'observables, à commencer, bien sûr, par les distributions habituelles de la physique statistique. La plus connue de ces distributions de maximum d'entropie exponentielle est la Gaussienne. Elle correspond au cas à une variable V, avec les 2 observables $f_1(V)=V$ et $f_2(V)=V^2$. La Gaussienne est donc la distribution de maximum d'entropie lorsque ce que l'on a retenu des expériences c'est leur moyenne expérimentale et celle de leur carré.

Appliquons ce type de raisonnement à nos 20 000 expériences.

Supposons que nous apprenions que nos 20 000 expériences sont les résultats de lancers consécutifs d'un dé. Force est de constater d'après les résultats expérimentaux que ce dé semble largement pipé (la loi de succession de Laplace obtenue est très différente de la distribution uniforme escomptée).

Un dé est un cube dans lequel de légères excavations ont été faites pour graver les chiffres sur les faces. Chacune de ces excavations provoque un léger déplacement du centre de gravité du dé en direction opposée. Une première hypothèse possible (C1) pour justifier le biais constaté consiste alors à dire que globalement le centre de gravité se déplace de manière à favoriser l'apparition des faces les plus légères, c'est-à-dire celles qui portent les chiffres les plus élevés. Pour tenter

d'observer cet effet escompté, nous choisissons l'observable $f_1(V)=V$. La moyenne expérimentale de $f_1(V)=$ vaut F1=3,5983. L'application du principe de maximum d'entropie nous conduit alors à une description du phénomène (une distribution de maximum d'entropie) ayant la forme :

$$P^*(v) = \frac{1}{Z} e^{-\lambda_1 v} \qquad [4.17]$$

avec $\lambda_1 = 3,372\ 10^{-2}$ et Z une constante de normalisation. Les probabilités correspondantes sont données dans la ligne C1 du tableau 2. Ces valeurs semblent encore très mal rendre compte des expériences.

Il est possible d'usiner 5 faces d'un dé sans avoir à le manipuler. Par contre, l'usinage de la dernière face suppose obligatoirement une manipulation et donc une perte de précision. On peut donc supposer qu'un dé a de bonne chance d'avoir un côté légèrement plus long (ou plus court) que les 2 autres. L'apparition des deux faces correspondantes sera alors légèrement moins probable (ou plus probable) que celle des 4 autres. On a donc 3 nouvelles hypothèses à tester :

- le côté reliant les faces 1 et 6 est de longueur différente (C2)
- le côté reliant les faces 2 et 5 est de longueur différente (C3)
- le côté reliant les faces 3 et 4 est de longueur différente (C4)

Prenons l'exemple de l'hypothèse C2, on peut essayer d'en quantifier l'observation avec l'observable $f_2(V)=0$ si V vaut 2, 3, 4 ou 5 et $f_2(V)=-1$ si V vaut 1 ou 6. La moyenne expérimentale de $f_2(V)$ vaut donc :

$$\frac{1}{20000} \sum_{i=1}^{6} n_i f_2(i) = -0,3589 \qquad [4.18]$$

L'application du principe de maximum d'entropie à cette hypothèse C2, nous conduit alors à une description du phénomène (une distribution de maximum d'entropie) ayant la forme:

$$P^*(V) = \frac{1}{Z} e^{-\lambda_1 V - \lambda_2 f_2(V)} \text{ avec } \lambda_1 = 3,234\ 10^{-2}, \lambda_2 = -1.104\ 10^{-1} \qquad [4.19]$$

Les probabilités correspondantes sont données dans la ligne C2 du tableau 1. Les mêmes calculs peuvent être effectués pour les hypothèses C3 et C4. Les résultats correspondants sont présentés dans le tableau 1 aux lignes C3 et C4.

	V=1	V=2	V=3	V=4	V=5	V=6	Total
n_i	3246	3449	2897	2841	3635	3932	20000
U Uniforme	0.16666	0.16667	0.16667	0.16667	0.16666	0.16667	1
L Laplace	0,16230	0,17245	0,14486	0,14206	0,18174	0,19659	1
C1	0,15294	0,15818	0,16361	0,16922	0,17502	0,18103	1
C2	0,16497	0,15259	0,15760	0,16278	0,16813	0,19393	1
C3	0,14803	0,16808	0,15843	0,16390	0,18612	0,17543	1
C4	0,16433	0,16963	0,14117	0,14573	0,18656	0,19258	1

Tableau 1

A la vue du tableau 1, l'hypothèse C4 semble rendre compte des données expérimentales beaucoup mieux que les autres. Cette intuition peut se vérifier mathématiquement en appliquant les techniques déjà rencontrées précédemment sous le nom de « problème inverse ». La règle [R1] nous donne:

$$P(C_i \mid D) = P(C_i)\frac{P(D \mid C_i)}{P(D)} \qquad [4.20]$$

Si nous supposons, *a priori*, que les 6 hypothèses en présence (U, L, C1, C2, C3 et C4) sont équiprobables, nous obtenons:

$$P(C_i \mid D) = \frac{P(D \mid C_i)}{\sum_{j=1}^{6} P(D \mid C_j)} \qquad [4.21]$$

$P(D|C_i)$ est donné par :

$$P(D \mid C_i) = W p_{i1}^{3246} p_{i2}^{3449} p_{i3}^{2897} p_{i4}^{2841} p_{i5}^{3635} p_{i6}^{3932} \qquad [4.22]$$

W (indépendant de Ci) est le nombre de permutations sur les 20000 expériences avec 3246 fois le résultat 1, 3449 fois le résultat 2, 2897 fois le résultat 3, 2841 fois le résultat 4, 3635 fois le résultat 5 et 3932 fois le résultat 6. P_{ij} est la probabilité d'obtenir le résultat j pour l'hypothèse Ci. En effectuant les calculs numériques on obtient :

Hypothèses	Laplace	C4	C2	C3	C1	U
P(Ci | D)	0.99	0.01	10^{-32}	10^{-35}	10^{-45}	10^{-59}

Tableau 2

L'hypothèse C4 est donc énormément plus probable que toutes les autres - sauf Laplace, qui se contente de coller aux résultats expérimentaux sans chercher à les « expliquer » .

L'hypothèse C4 ayant une interprétation et une justification physique, on est libre de considérer qu'il vaut mieux l'adopter comme description que Laplace[38]. C'est un choix subjectif, qui peut se traduire mathématiquement en lui donnant une probabilité *a priori* P(C4) beaucoup plus grande qu'à la loi de succession de Laplace.

En fait, les données utilisées dans cet exemple sont des données réelles issues d'une expérience menée par Wolf il y a plus d'un siècle. Six nombres (les nombres de fois où chacun des 6 résultats possibles est apparu), quelques connaissances préalables traduisant un peu de bon sens physique, et le principe de maximum d'entropie : voilà qui nous permet de penser avec une très grande confiance que ce dé que nous n'avons évidemment jamais vu, était légèrement oblong suivant l'axe « 3-4 ».

B.3.2 Justification du principe de maximum d'entropie

La justification la plus intuitive du principe de maximum d'entropie repose sur un argument combinatoire issu directement de son origine venant de la mécanique statistique. Cet argument fut proposé originellement par Boltzmann.

Supposons que nous ayons un ensemble de n particules identiques et que chacune de ces particules puisse être dans q différents états microscopiques équiprobables.

Définissons ν_k, un état macroscopique, comme un ensemble $\{n_1,...,n_q\}$ de q nombres tel que n_i soit le nombre de particules dans l'état microscopique i.

Nous devons, bien sûr, avoir :

$$\sum_{i=1}^{q} n_i = n \qquad [4.23]$$

et de plus, par exemple, des contraintes telles que :

$$\sum_{i=1}^{q} n_i e_i = e \qquad [4.24]$$

où, e désigne l'énergie globale du système et e_i l'énergie de l'état i.

---

[38] Ce genre de connaissance préalable ayant un sens physique permet en général une meilleure généralisation que la loi de succession de Laplace.

De plus, cela présente l'immense avantage pratique de conduire à des représentations beaucoup plus compactes : m (nombre d'observables) paramètres sont suffisants, au lieu des $<\Omega>$ nécessaires avec Laplace.

Appelons $W(\nu_k)$ le nombre de manière de réaliser l'état macroscopique ν_k. Nous avons:

$$W(\upsilon_k) = \frac{n!}{n_1! \times n_2! \times ... \times n_q!} \quad [4.25]$$

Pour Boltzmann, l'état macroscopique le plus probable est alors celui qui peut être réalisé du plus grand nombre de manières microscopiques possible, c'est-à-dire celui qui maximise $W(\nu_k)$ tout en respectant les contraintes imposées [4.23] et [4.24].

En utilisant la formule de Stirling :

$$\log(n!) = n\log(n) - n + \sqrt{2\pi n} + \frac{1}{12n} + o\left(\frac{1}{n^2}\right) \quad [4.26]$$

on voit que l'état le plus probable est celui qui maximise (toujours en respectant les contraintes [4.23] et [4.24]) :

$$log\left(W(\upsilon_k)\right) \approx -n\sum_{i=1}^{q} \frac{n_i}{n} log\left(\frac{n_i}{n}\right) \quad [4.27]$$

On retrouve ainsi la forme de la fonction H.

Dans notre terminologie PaL, le principe de maximum d'entropie peut être vu comme l'exact équivalent du raisonnement de Boltzmann.

Supposons que dans notre système formel probabiliste, nous ayons q propositions mutuellement exclusives possibles (l'analogue des états microscopiques).

Supposons que nous ayons n expériences à notre disposition, une expérience consistant à constater que l'une des q propositions est vérifiée (l'analogue d'une particule étant dans l'état q).

Définissons δ_k, une description, comme une distribution de probabilité :

$$\left\{p_1, ..., p_q\right\} = \left\{\frac{n_1}{n}, ..., \frac{n_q}{n}\right\} \quad [4.28]$$

sur les q propositions mutuellement exclusives du système formel (l'analogue des états macroscopiques).

Nous devons, bien sur, avoir :

$$\sum_{i=1}^{q} p_i = 1 \quad [4.29]$$

Nous pouvons de plus avoir des contraintes traduisant les connaissances préalables d'observations que nous avons sur le phénomène. Ces contraintes peuvent prendre des formes très diverses, le principe de maximum entropie étant toujours applicable. Par exemple, les connaissances préalables peuvent avoir la forme de m observables :

$$\forall j, j \in \{1, ..., m\}; \sum_{i=1}^{q} p_i f_j(i) = F_j \quad [4.30]$$

Appelons $W(\delta_k)$ le nombre de manières de permuter les expériences en conservant les fréquences n_i/n correspondant à la description δ_k. Nous avons:

$$W(\delta_k) = \frac{n!}{n_1! \times n_2! \times \ldots \times n_q!} \qquad [4.31]$$

En utilisant la formule de Stirling, on obtient comme précédemment que la description la meilleure est celle qui correspond au plus grand nombre possible de permutations des expériences, c'est-à-dire celle qui maximise (en respectant les m+1 contraintes issues de [4.29] et [4.30]):

$$log(W(\delta_k)) \approx -n \sum_{i=1}^{q} p_i \, log(p_i) \qquad [4.32]$$

Reprenons l'exemple de l'hypothèse C1. La variable V peut prendre 6 valeurs entières entre 1 et 6. Nous avons fait 20 000 expériences dont nous avons choisi de ne mémoriser que la moyenne (observable $f_1(V)=V$) valant 3,5983.

Parmi les 6^{20000} séries d'expériences possibles, très peu, bien sûr, ont cette moyenne. Ces dernières peuvent être regroupées en « classes » (les *descriptions*), chaque classe étant caractérisée par la donné de 6 nombres (les p_i ou n_i) correspondant aux nombres d'apparitions de chacune des 6 valeurs possibles. A une classe donnée correspond $W(\delta_k)$ séries d'expériences possibles.

Numériquement, soit les trois classes :
- $\delta 1$: (3058, 3165, 3272, 3384, 3500, 3621)
- $\delta 2$: (3058, 3265, 3172, 3284, 3600, 3621)
- $\delta 3$: (0, 0, 8034, 11966, 0, 0)

ayant toutes les trois une moyenne de 3,5983, on calcule aisément (à partir de la formule [4.32]) que la première peut être réalisée par 10^{15549} séries d'expériences, la deuxième par 10^{15546} séries d'expériences et la troisième par 10^{5869} séries d'expériences. *Donc en tirant au hasard parmi les séries de 20000 expériences ayant 3,5983 comme moyenne (ce que nous pensons avoir fait en observant nos 20000 données) nous avons 1000 fois plus de chance de tirer une série d'expériences de la classe $\delta 1$ que de la classe $\delta 2$ et 10^{9680} fois plus de chance de tirer une série d'expériences de la classe $\delta 1$ que $\delta 3$. Comment, dès lors, ne pas considérer que la classe $\delta 1$ est une meilleure description que les 2 autres ?*

Les connaissances préalables définissent « l'instrument d'optique », le point de vue sous lequel on décide d'observer le phénomène étudié. Différentes connaissances préalables correspondent à différents points de vue et résultent en des distributions de maximum d'entropie, des *descriptions* différentes. Changer de connaissances préalables, c'est changer la forme de la paroi de la caverne de Platon sur laquelle se projettent les ombres portées de la « réalité », c'est changer inexorablement la perception intime du phénomène observé.

Il existe, bien entendu, des manières plus rigoureuses de justifier l'emploi du principe de maximum d'entropie, notamment les théorèmes, dits de « concentration d'entropie », tels que démontrés par Jaynes et par Robert. On se référera aux notes bibliographiques pour plus de précisions.

B.4 Notes bibliographiques

La référence incontournable concernant ce paragraphe est le livre de Jaynes intitulé « Probability Theory: The Logic of Science » (Jaynes95). Cet ouvrage inachevé n'a pas encore été publié mais est accessible électroniquement depuis déjà plus de deux ans et s'enrichit régulièrement de nouvelles parties. Ce livre est le « chef-d'œuvre » de Jaynes qui fait la synthèse des travaux de toute une vie concernant le raisonnement probabiliste, les techniques de maximum d'entropie et leurs applications à la physique.

Un historique passionnant de cette approche peut être trouvé dans (Jaynes79).

Une analyse épistémologique des probabilités est donnée dans (Matalon67) précisant notamment la nature du débat entre « fréquentistes » et « subjectivistes ».

Concernant plus spécifiquement le théorème de Cox on se référera à (Cox46) et (Cox61). Une démonstration de ce théorème peut aussi être trouvée au chapitre 2 de (Jaynes95).

La logique, comme cas particulier des probabilités, est un sujet abordé par plusieurs auteurs, notamment, bien sur, par Jaynes dans (Jaynes95). (Cox79) est un article entièrement consacré à ce sujet.

Les problèmes directs et inverses sont largement développés dans les chapitres 3, 4, 6, 9 et 10 de (Jaynes95).

Pour une introduction à la notion d'entropie, on se référera bien évidemment à l'article fondateur de Shannon (Shannon49) qui introduisit le premier la notion d'entropie en dehors de la thermodynamique et de la mécanique statistique, ainsi qu'au passionnant livre de Campbell (Campbell82) qui présente bien les différents aspects de cette notion.

Les justifications théoriques de l'approche probabiliste et du principe de maximum d'entropie peuvent être trouvées, notamment, dans (Cox46), (Cox61), (Cox79), (deFinetti72), (Shore80), (Shore81), (VanCampenhout81), (Jaynes82), (Cheeseman85), (Shore86), (Hunter86) et (Robert91). (Jaynes82) et (Robert91) méritant une mention particulière puisqu'ils proposent des démonstrations des théorèmes de concentration d'entropie, respectivement dans les cas discret et continu.

Les outils mathématiques correspondant à ces techniques et les applications en physique sont amplement décrits dans (Levine79), (Smith85), (Tarentola87), (Erickson88a), (Erickson88b), (Bretthorst88), (Kapur92), (Mohammad-Djafari92) et évidemment (Jaynes95).

L'exemple du dé de Wolf est présenté et discuté en détail dans (Fougere88).

Des exemples d'applications à des problèmes de contrôle sont donnés par (MacKay92) ou (Neal93).

La description des « réseaux bayésiens », intéressante implantation informatique du raisonnement probabiliste, peut être trouvée dans (Pearl91) même si la dimension apprentissage en est presque totalement absente.

B.5 Bibliographie

Bessière, P.; *LOLITA : Un langage de programmation logique intégré à LISP*; Cahier technique I.T.M.I., 1987.

Campbell Jerremy ; *Gramatical Man* ; Simon & Schuster, 1982

Cheeseman P. ; *In defense of Probability* ; Proceedings of AAAI85, 1985

Colmerauer A. ; *Theoretical model of PROLOG II* ; Logic Programming and its Applications; Ablex Publishing Corporation; 1986

Cox R. T. ; *Probability, Frequency, and Reasonable Expectation* ; American Journal of Physic, N°17 ; 1946

Cox R. T. ; *The algebra of probable inference* ; The John Hopkins Press, Baltimore, 1961

Cox R. T. ; *Of inference and inquiry, an essay in inductive logic ;* in *The maximum entropy formalism*, edited by Raphael D. Levine & Myron Tribus ; M.I.T. Press, 1979

de Finetti B. ; *Probability, induction and statistics* ; John Willey & sons, 1972

Erickson Gary J. & Smith C. Ray ; *Maximum-Entropy and Bayesian methods in science and engineering* ; *Volume 1 : Foundations* ; Kluwer Academic Publishers ; 1988

Erickson Gary J. & Smith C. Ray ; *Maximum-Entropy and Bayesian methods in science and engineering* ; *Volume 2 : Applications* ; Kluwer Academic Publishers ; 1988

Fougere P. F. ; *Maximum entropy calculations on a discrete probaility space* ; in *Maximum entropy and bayesian methods in science and engineering*, Vol. 1, edited by G.J. Erickson & C.R. Smith ; Kluewer Academic Publishers ; 1988

Hunter Daniel ; *Uncertain reasoning using Maximum entropy Inference* in *Uncertainty in Artificial Intelligence* ; edited by L. N. Kanal & J. F. Lemmer ; Elsevier Science Publishers, 1986

Jaynes E.T. ; *Where do we Stand on Maximum Entropy?* ; in *The maximum entropy formalism*, edited by Raphael D. Levine & Myron Tribus ; M.I.T. Press, 1979

Jaynes E. T. ; *On the rationale of maximum-entropy methods ;* Proceedings of the IEEE, 1982

Jaynes E.T. ; *Probability theory - The logic of science* ; à paraître. Version provisoire disponible à http://bayes.wustl.edu. ; 1995

Kapur J.N. & Kesavan H.K. ; *Entropy optimization principles with applications* ; Academic Press ; 1992

Levine Raphael D. & Tribus Myron ; *The maximum entropy formalism* ; MIT Press ; 1979

MacKay David J.C. ; *A practical bayesian framework for backprop networks* ; Neural computation ; 1992

Matalon Benjamin ; *Epistémologie des probabilités* ; dans *Logique et connaissance scientifique* sous la direction de Jean Piaget ; Encyclopédie de la Pléiade ; Editions Gallimard ; Paris, France ; 1967

Mohammad-Djafari A. & Demoment G. ; *Maximum entropy and bayesian methods* ; Kluwer Academic Publishers ; 1992

Neal Radford M. ; *Probabilistic inference using Markov chain Monte-Carlo Methods* ; Technical Report university of Toronto ; 1993

Pearl Judea ; *Probabilistic reasoning in intelligent systems : Networks of plausible inference* ; Morgan Kaufmann Publishers, San Mateo, California, USA ; 1991

Robert Claudine ; *An entropy concentration theorem: applications in artificial intelligence and descriptive statistics* ; Journal of Applied Probabilities ; september 1990

Robert Claudine ; *Modèles statistiques pour l'intelligence artificielle : l'exemple du diagnostic médical* ; Masson, Paris, France ; 1991

Robinson J.A. & Sibert E.E. ; *LOGLISP : an alternative to PROLOG*; Machine Intelligence; Vol. 10; 1983

Shannon Claude E. ; *The mathematical theory of communication* ; University of illinois, 1949

Shore J. E. & Johnson R. W. ; *Axiomatic derivation of the principle of Maximum Entropy and the Principle of Minimum Cross-Entropy ;* IEEE Transactions on Information Theory, 1980

Shore J. E. & Johnson R. W. ; *Properties of cross-entropy minimization ;* IEEE Transactions on Information Theory, 1981

Shore John E. ; *Relative entropy, probabilistic inference and A.I.* in *Uncertainty in Artificial Intelligence* ; edited by L. N. Kanal & J. F. Lemmer ; Elsevier Science Publishers, North-Holland, 1986

Smith C. Ray & Grandy, W. T. Jr. ; *Maximum-Entropy and bayesian methods in inverse problems* ; D. Reidel Publishing Company ; 1985

Tarentola Albert ; *Inverse Problem Theory ; Methods for data fitting and model parameters estimation* ; Elsevier ; New York, USA ; 1987

Van Campenhout J. M. & Cover T.M. ; *Maximum entropy and conditional probability ;* IEEE Transactions on Information Theory, 1981

RÉSUMÉ. Cette thèse propose une méthode originale de programmation de robot fondée sur l'inférence et l'apprentissage bayésien. Cette méthode traite formellement des problèmes d'incertitude et d'incomplétude inhérents au domaine considéré. En effet, la principale difficulté de la programmation des robots viens de l'inévitable incomplétude des modèles utilisés. Nous exposons le formalisme de description d'une tâche robotique ainsi que les méthodes de résolutions. Ce formalisme est inspiré de la théorie du calcul des probabilités, proposée par le physicien E. T. Jaynes : « Probability as Logic ». L'apprentissage et les techniques de maximum d'entropie traduisent l'incomplétude en incertitude. L'inférence bayésienne offre un cadre formel permettant de raisonner avec cette incertitude. L'apport principal de cette thèse est la définition d'un système générique de programmation pour la robotique et son application expérimentale. Nous l'illustrons en utilisant ce système pour programmer une application de surveillance pour un robot mobile : le Khepera. Pour cela, nous utilisons des ressources génériques de programmation appelées « descriptions ». Nous montrons comment définir et utiliser de manière incrémentale ces ressources (comportements réactifs, fusion capteur, reconnaissance de situations et séquences de comportements) dans un cadre systématique et unifié. Nous discutons des différents avantages de notre approche : expression des connaissances préalables, prise en compte et restitution de l'incertitude, programmation directe et inverse. Nous proposons des perspectives à ce travail : choix d'architecture et planification. Nous situons notre travail dans un cadre épistémologique plus vaste en opposant, dans le cadre de la robotique autonome, l'approche « classique » relevant de la « cognition de haut niveau » et l'approche « réactive » associée à une « cognition de bas niveau ». Nous montrons finalement comment nos travaux proposent de faire le lien entre ces deux extrêmes.

Mots-clés : science cognitive, robotique autonome, programmation des robots, inférence bayésienne, ancrage perceptif des symboles, incertitude.

ABSTRACT. This thesis proposes an original method for robotic programming based on bayesian inference and learning. This method formally deals with problems of uncertainty and incomplete information that are inherent to the field. Indeed, the principal difficulties of robot programming comes from the unavoidable incompleteness of the models used. We present the formalism for describing a robotic task as well as the resolution methods. This formalism is inspired by the theory of the probability calculus, suggested by the physicist E T Jaynes: "Probability as Logic". Learning and maximum entropy principle translates incompleteness into uncertainty. Bayesian inference offers a formal framework for reasoning with this uncertainty. The main contribution of this thesis is the definition of a generic system of robotic programming and its experimental application. We illustrate it by programming a surveillance task with a mobile robot: the Khepera. In order to do this, we use generic programming resources called "descriptions". We show how to define and use these resources in an incremental way (reactive behaviors, sensor fusion, situation recognition and sequences of behaviors) within a systematic and unified framework. We discuss the various advantages of our approach: statement of preliminary knowledge, taking into account uncertainty, direct and inverse programming. We suggest perspectives for our work: choice of architecture and planning. We place our work within a wider epistemological horizon while opposing, within the framework of autonomous robotics, the "traditional" approach concerning "high level cognition" and the "reactive" approach associated with the "low level cognition". We finally show how our work proposes to establish a link between these two extremes.

Keywords : cognitive science, autonomous robotic, robot programming, bayesian inference, symbol grounding problem, uncertainty.