Apprentissage Interactif de Liaisons Directes entre Perceptions Visuelles et Actions

Interactive Learning of Direct Mappings between Visual Percepts and Actions

Sébastien Jodogne* Justus H. Piater

Institut Montefiore (B28)
Université de Liège
B-4000 Liège, Belgique
{S.Jodogne,Justus.Piater}@ULg.ac.be

Résumé

Nous introduisons des algorithmes capables d'apprendre des liaisons directes entre stimuli visuels et actions. L'apprentissage se fait de manière totalement autonome, par le biais d'interactions avec l'environnement. Ces algorithmes consistent à introduire un classificateur d'images avant un algorithme d'Apprentissage par Renforcement. Fondé sur le principe de l'apparence locale, ce classificateur partitionne l'espace visuel en fonction de la présence ou de l'absence de descripteurs locaux à haut contenu informationnel. Le classificateur d'images est progressivement affiné par la sélection de nouveaux descripteurs locaux lorsque de l'ambiguïté perceptuelle est détectée. Ceci nous permet de réduire la haute dimensionnalité d'un espace visuel à un domaine traitable par les algorithmes d'Apprentissage par Renforcement. Il devient ainsi possible d'apprendre des mises en correspondance directes entre images et actions de manière interactive. Les résultats expérimentaux sur une tâche de navigation visuelle dans un espace continu illustre l'applicabilité de notre approche.

Mots Clef

Apprentissage Visuel, Apparence Locale, Sélection de Caractéristiques.

Abstract

We introduce flexible algorithms that can automatically learn direct mappings from visual stimuli to actions. The learning process is carried on in a fully autonomous fashion, through interactions with the environment. Our algorithms work by introducing an image classifier in front of a Reinforcement Learning algorithm. The classifier is founded on the local-appearance paradigm: It partitions the visual space according to the presence or absence of highly informative local descriptors. The image classifier is incrementally refined by selecting new local descriptors whenever perceptual aliasing is detected. Thus, we reduce the visual input domain down to a size manageable by Reinforcement Learning, permitting us to learn direct percept-to-action mappings. Experimental results on a continuous visual navigation task illustrate the applicability of the framework.

Keywords

Visual Learning, Local Appearance, Feature Selection.

1 Introduction

De nombreuses tâches robotiques réactives peuvent être résolues par des algorithmes en boîte noire qui connectent les différentes perceptions possibles à l'action la mieux adaptée. En général, il est difficile de mettre manuellement en correspondance un domaine d'entrée et un domaine de sortie. La tâche devient très ardue si les perceptions contiennent des images, car les domaines visuels sont bruités et à haute dimensionnalité. Cet article introduit des algorithmes capables de réaliser des mises en correspondance « images-vers-actions » par le biais d'un protocole d'apprentissage souple et entièrement automatique.

La neuropsychologie suggère que les êtres humains apprennent à extraire les informations pertinentes hors de données visuelles de manière *interactive*, sans supervision externe [5]. En évaluant les conséquences de nos actions sur notre environnement, nous apprenons à isoler les informations visuelles qui sont importantes pour résoudre une tâche donnée. Ce processus d'apprentissage dépend de la tâche, puisque deux tâches distinctes ne doivent pas nécessairement faire les mêmes distinctions visuelles.

*Aspirant du Fonds National de la Recherche Scientifique Belge (F.N.R.S)

Ces observations ont donné naissance à la théorie de l'*Apprentissage par Renforcement* (*Reinforcement Learning*, RL) [2, 23]. RL est une théorie générale permettant de modéliser le comportement d'un agent qui apprend une mise en correspondance perceptions-vers-actions à travers ses interactions avec l'environnement. À aucun moment un superviseur externe ne montre à l'agent les réactions souhaitées. Au lieu de cela, quand l'agent fait une bonne ou une mauvaise action, son environnement le félicite ou le punit, en lui donnant un *signal de renforcement*. RL compte trois avantages majeurs : son caractère totalement automatique, le fait qu'il est biologiquement motivé et les hypothèses très lâches qu'il pose sur l'environnement. Le prix à payer pour cette flexibilité est de mauvaises performances quand l'espace perceptuel de l'agent est bruité ou à haute dimensionnalité. Ce problème provient surtout du fait que les algorithmes de RL utilisent les perceptions brutes de l'agent, telles que fournies par ses capteurs. Malheureusement, ceci empêche l'usage de RL dans des domaines d'entrée visuels.

Toutefois, afin de prendre la bonne décision, l'agent peut souvent se contenter de focaliser son attention sur des motifs robustes et hautement distinctifs présents dans ses perceptions. Il s'agit en fait du postulat qui a donné naissance aux méthodes dites d'*apparence locale* dans le contexte de la vision par ordinateur. Ces méthodes ont prouvé leur intérêt dans des applications telles que la mise en correspondance d'images, la recherche d'images dans une base de données ou la reconnaissance d'objets [10, 18]. Elles résident sur la détection de discontinuités dans le signal visuel grâce à des *détecteurs de points d'intérêt* [19]. Les similarités entre images sont ensuite identifiées en utilisant une *description locale* [12] des voisinages des points d'intérêts dans les images. De telles techniques sont à la fois puissantes et souples, puisqu'elles ne requièrent aucune segmentation, aucun modèle 3D et sont robustes aux occlusions partielles.

Il semble dès lors prometteur de combiner Apprentissage par Renforcement et méthodes d'apparence locale afin de pouvoir générer des correspondances images-vers-actions. Ceci nous a motivés à introduire, avant l'algorithme de RL, un classificateur d'images qui traduit les images fournies par les capteurs en une classe symbolique, selon la présence de certains descripteurs locaux à haut contenu informationnel détectés aux points d'intérêt des images. Cette étape de pré-traitement a pour objectif de réduire la taille du domaine d'entrée de l'algorithme de RL. La vitesse de convergence, les capacités de généralisation et la robustesse au bruit de RL en sont ainsi améliorées. La difficulté centrale de cette approche est la sélection dynamique des descripteurs locaux ayant un fort pouvoir discriminant.

Nous proposons l'algorithme suivant. Initialement, l'agent ne connaît qu'une seule classe symbolique, si bien que le classificateur d'images amène toutes les images dans cette classe. Évidemment, ceci introduit une forme d'*ambiguïté perceptuelle* (*perceptual aliasing*) [26] : l'agent ne peut pas toujours prendre la bonne décision, puisque des perceptions nécessitant des réactions différentes sont associées à la même classe symbolique. Ensuite, l'agent détecte les classes visuelles ambiguës. Puisqu'il ne peut compter sur aucun superviseur externe, l'agent doit, pour ce faire, se baser uniquement sur une analyse statistique des renforcements perçus. Quand une classe chargée d'ambiguïté est détectée, l'agent sélectionne un nouveau descripteur local qui est *distinctif*, c'est-à-dire qui explique le mieux possible les variations qui ont été observées dans les renforcements pour cette classe visuelle. Ce nouveau descripteur est finalement utilisé pour affiner le classificateur d'images. L'agent intègre ainsi dans son classificateur d'images de nouveaux descripteurs jusqu'à la disparition des ambiguïtés.

Cette manière itérative de procéder est très différente des techniques classiques de discrétisation utilisées dans le cadre de l'Apprentissage par Renforcement [2]. Celles-ci supposent en effet que les caractéristiques intéressantes (i.e., les descripteurs locaux distinctifs), ou bien leur nombre, sont connus à l'avance.

L'article est structuré comme suit. D'abord, nous rappelons les résultats fondamentaux de la Programmation Dynamique et de l'Apprentissage par Renforcement. Ensuite, nous décrivons notre algorithme en détail. Les résultats expérimentaux sont ensuite présentés.

2 Éléments Théoriques

2.1 Modéliser l'Environnement et l'Agent

En Apprentissage par Renforcement[1], l'environnement est généralement modélisé sous la forme d'un *Processus Décisionnel de Markov* (*Markov Decision Process*, MDP). Un MDP est un tuple $\langle S, A, \mathcal{T}, \mathcal{R} \rangle$, où :

- S est un ensemble fini d'états,
- A est un ensemble fini d'actions,
- $\mathcal{T}$ est une *fonction de transition* probabiliste de $S \times A$ vers S, et
- $\mathcal{R}$ est une *fonction de renforcement* de $S \times A$ vers $\mathbb{R}$.

Un MDP suit la dynamique discrète suivante : si au temps t, l'agent choisit l'action a_t alors que l'environnement se trouve dans un état s_t, l'agent perçoit un renforcement numérique $r_{t+1} = \mathcal{R}(s_t, a_t)$ et, au temps $t+1$, l'état de l'environnement sera s_{t+1} avec une probabilité donnée par $\mathcal{T}(s_t, a_t, s_{t+1})$.

En général, l'agent n'a pas un accès direct à l'état s_t dans lequel se trouve l'environnement. Au lieu de cela, il perçoit l'état courant par le biais de ses capteurs. Définissons

[1] Dans cet article, nous adoptons les suggestions de terminologie formulées sur le site Internet `http://www.loria.fr/projets/PDMIA/index.php?sub=outils` pour la traduction des termes anglais de l'Apprentissage par Renforcement. Nous conservons les acronymes anglais par soucis de cohérence avec la littérature anglophone.

l'*espace perceptuel* P comme étant l'ensemble des perceptions que les capteurs de l'agent peuvent retourner. Dans le cas de tâches visuelles, P est un ensemble d'images. Dès lors, du point de vue de l'agent, une *interaction* avec son environnement est définie comme un quadruplet $\langle p_t, a_t, r_{t+1}, p_{t+1} \rangle$, où p_t (resp. p_{t+1}) est la perception fournie par les capteurs de l'agent en présence de l'état s_t (resp. s_{t+1}).

Si toutes les perceptions peuvent être générée par au plus un état, l'espace perceptuel est dit *totalement observable* et l'agent est capable de distinguer les états de l'environnement en utilisant uniquement ses capteurs. Dans le cas contraire, l'espace perceptuel est dit *partiellement observable*. Dans la suite de cet article, nous ne considérerons que des espaces perceptuels totalement observables.

Une *liaison perceptions-vers-actions* est une fonction probabiliste invariante au cours du temps $\pi : P \mapsto A$ reliant les perceptions aux actions. Une liaison perceptions-vers-actions donne à l'agent de la probabilité avec laquelle il devrait choisir une action en présence d'une perception donnée. Dans la terminologie RL, une telle liaison est appelée une *politique de contrôle Markovienne et stationnaire*.

2.2 Programmation Dynamique

Le *problème décisionnel de Markov* pour un MDP fixé consiste à trouver une liaison perceptions-vers-actions qui maximise une évaluation des performances de l'agent au cours du temps. Cette fonction d'évaluation est calculée en fonction des renforcements perçus et est appelée *fonction-objectif*.

La fonction-objectif la plus souvent utilisée en RL est le *retour actualisé*. Étant donnée une suite infinie d'interactions $\langle p_t, a_t, r_{t+1}, p_{t+1} \rangle$ (avec $t = 0, 1, \ldots$), le retour actualisé au temps t est défini par :

$$R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i+1}, \tag{1}$$

où $\gamma \in [0, 1[$ est le *coefficient d'actualisation* donnant la valeur courante des renforcement futurs. Par conséquent, quand le coefficient d'actualisation est non nul, l'agent ne doit pas seulement maximiser ses renforcements immédiats (i.e., la suite des r_t), mais doit aussi veiller à prendre des actions éventuellement peu intéressantes dans l'immédiat, mais qui l'amèneront dans des portions de l'espace d'états où il pourra obtenir de plus importantes récompenses. Par exemple, dans le jeu d'échecs, le sacrifice d'une pièce induit un mauvais renforcement immédiat, mais peut mener au gain de la partie. Ce compromis est appelé *problème de la récompense tardive* (*delayed-reward problem*).

Les MDP peuvent être résolus en utilisant la théorie de la *Programmation Dynamique* (*Dynamic Programming*, DP). Soit une liaison perceptions-vers-actions π. Appelons $Q^\pi(s, a)$, la *fonction de qualité de la liaison* π, donnant pour chaque état $s \in S$ et chaque action $a \in A$, le retour actualisé moyen attendu quand l'agent démarre de l'état s, exécute l'action a et réagit ensuite en fonction de π :

$$Q^\pi(s, a) = \mathrm{E}^\pi \left\{ R_t \mid s_t = s,\ a_t = a \right\}, \tag{2}$$

où E^π désigne la valeur moyenne si l'agent suit la liaison π sur un temps infini. Définissons aussi la *transformée* H des fonctions de qualité vers les fonctions de qualité comme suit :

$$(HQ)(s, a) = \mathcal{R}(s, a) + \gamma \sum_{s' \in S} \mathcal{T}(s, a, s') \max_{a' \in A} Q(s', a'), \tag{3}$$

pour tout $s \in S$ et $a \in A$.

Bellman a montré que toutes les liaisons perceptions-vers-actions optimales pour un MDP donné partagent la même fonction de qualité, dénotée Q^* [1]. Cette fonction de qualité optimale existe toujours et satisfait l'équation dite de Bellman :

$$HQ^* = Q^*. \tag{4}$$

Une fois que Q^* est connue, une liaison perceptions-vers-actions optimale π^* peut être aisément obtenue en posant :

$$\pi^*(s) = \operatorname*{argmax}_{a \in A} Q^*(s, a), \tag{5}$$

pour tout état $s \in S$. La Programmation Dynamique permet dès lors de réduire un problème décisionnel de Markov à la résolution du système d'équations non linéaire (4).

En pratique, on utilise plutôt des méthodes incrémentales afin de résoudre les équations de Bellman. Ainsi, un des algorithmes standards de Programmation Dynamique, connu sous le nom d'*Itération sur la Valeur État-Action* (*State-Action Value Iteration*), profite du fait que H est une *transformée de contraction*, i.e. qu'il existe un scalaire $\rho < 1$ tel que $||HQ - HQ'||_\infty \leq \rho ||Q - Q'||_\infty$ pour toute paire de fonctions de qualité (Q, Q'). En vertu du théorème de Banach pour les points fixes (cf. par exemple [4]), la suite $\hat{Q}_{k+1} = H\hat{Q}_k$ (où $k = 0, 1, 2, \ldots$) converge vers Q^*, quelque soit $\hat{Q}_0$, par rapport à la métrique $|| \cdot ||_\infty$.

2.3 Apprentissage par Renforcement

L'*Apprentissage par Renforcement* (*Reinforcement Learning*, RL) [2, 23] est un ensemble de méthodes algorithmiques pour résoudre des problèmes décisionnels de Markov quand le MDP sous-jacent est inconnu. L'entrée des algorithmes de RL est uniquement une suite d'interactions $\langle p_t, a_t, r_{t+1}, p_{t+1} \rangle$ de l'agent avec son environnement. Les techniques de RL sont souvent divisées en deux catégories :

– les *méthodes basées sur un modèle*, qui construisent d'abord une estimation du MDP sous-jacent (e.g. en calculant les fréquences relatives qui apparaissent dans la suite d'interactions), puis qui utilisent les algorithmes classiques de la Programmation Dynamique pour extraire une liaison perceptions-vers-actions optimale ;

– les *méthodes libres de modèle*, qui ne construisent pas une estimation du MDP sous-jacent, telles que $SARSA$, $TD(\lambda)$ [23] et le populaire Q *Learning* [25].

3 Apprendre des Liaisons Visuelles

3.1 Classificateur d'Images

Comme précisé dans l'introduction, nous nous proposons d'introduire un classificateur d'images entre les capteurs de l'agent et un algorithme d'Apprentissage par Renforcement. Le classificateur d'images va être progressivement affiné. À l'étape k de notre algorithme, le classificateur, qui sera dénoté $\mathcal{C}_k$, partitionne l'espace perceptuel P en un nombre fini m_k de *classes visuelles* $\{V_{k,1}, \ldots, V_{k,m_k}\}$.

Le processus de classification est fait en accord avec le paradigme de l'*apparence locale*, qui consiste à fixer l'attention de l'agent sur des descripteurs locaux hautement distinctifs détectés aux points d'intérêt des images d'entrée. Formellement, appelons F un ensemble potentiellement infini de descripteurs locaux [12]. En général, F correspond à $\mathbb{R}^n$ pour un certain naturel n. Les éléments de F seront indifféremment appelés descripteurs locaux ou caractéristiques visuelles. Nous supposons l'existence d'un *détecteur de caractéristiques visuelles*, qui est une fonction booléenne $\mathcal{F} : P \times F \mapsto \mathcal{B}$ testant s'il existe, dans l'image donnée en argument, un *point d'intérêt* [19] dont la description du voisinage correspond au descripteur local fourni en argument. En général, on utilisera un seuillage sur la distance euclidienne ou sur la métrique de Mahalanobis pour tester l'égalité de deux descripteurs locaux.

Le but de nos algorithmes est de générer une suite de classificateurs d'images de plus en plus fins. Le premier classificateur $\mathcal{C}_0$ attribue toutes les images d'entrée à la même classe visuelle $V_{0,1}$. Chaque classificateur est obtenu à partir de son prédécesseur en lui ajoutant de nouvelles caractéristiques visuelles.

Du fait de l'incrémentalité de cette construction, il est naturel d'implémenter de tels classificateurs sous la forme d'arbres de décision binaires. Les classes visuelles correspondent aux feuilles des arbres. Chaque nœud interne est étiqueté par le descripteur local dont il faut tester la présence en ce nœud. Pour classifier une image, le système démarre du nœud racine et progresse vers le haut de l'arbre jusqu'à atteindre une feuille, en accord avec les résultats du détecteur de caractéristiques $\mathcal{F}$ pour chaque descripteur local trouvé durant la montée. Pour affiner un classificateur en utilisant une nouvelle caractéristique visuelle, il suffit de remplacer la feuille correspondant à la classe visuelle à affiner par un nœud interne qui teste la présence ou l'absence de cette caractéristique.

3.2 Architecture d'Apprentissage

Nous sommes intéressés par la construction d'une liaison images-vers-actions $\pi : P \mapsto A$. À l'étape k, étant donnée une perception visuelle $p \in P$, le résultat de $\mathcal{C}_k(p)$ est fourni comme entrée à l'algorithme de RL, au lieu de l'image brute p. Cette architecture d'apprentissage sera appelée par la suite *Apprentissage par Renforcement de Classes Visuelles* (*Reinforcement Learning of Visual Classes*, RLVC) [7, 8]. Elle est illustrée à la figure 1.

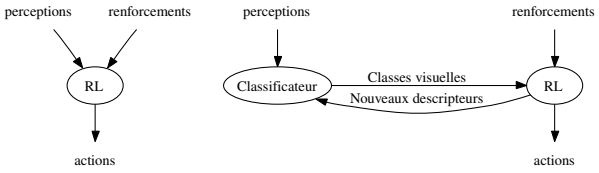


FIG. 1 – Comparaison des flux d'informations entre l'Apprentissage par Renforcement classique et notre approche.

RLVC consiste en deux processus d'apprentissage entrelacés : (i) la sélection dynamique des caractéristiques visuelles distinctives et (ii) l'Apprentissage par Renforcement d'une liaison entre les classes visuelles et les actions. Les deux défis majeurs posés par RLVC sont par conséquent (i) l'identification des classes visuelles qui doivent être affinées et (ii) la sélection de nouvelles caractéristiques qui conviennent pour réaliser cet affinage.

3.3 Ambiguïtés Perceptuelles

Avant que l'agent n'ait construit un classificateur suffisamment fin, des stimuli visuels nécessitant des réactions différentes peuvent être confondus dans la même classe visuelle. Dans ce cas, les bonnes décisions ne peuvent pas toujours être prises par l'algorithme de RL sur base de ses entrées. Du point de vue de l'algorithme de RL, l'espace perceptuel est seulement partiellement observable (cf. section 2.1). Ce phénomène est connu sous le nom d'*ambiguïté perceptuelle* (*perceptual aliasing* ou *hidden state*). Dès lors, détecter si un affinage est nécessaire pour une classe visuelle se ramène à déterminer si de l'ambiguïté perceptuelle se manifeste dans cette classe.

Deux solutions au problème générique de l'ambiguïté perceptuelle ont été proposées dans la littérature. Soit l'agent identifie puis évite les états perceptuellement ambigus [26] ; soit l'agent gère une mémoire à court terme qui lui permet de lever les ambiguïtés sur ses perceptions [3, 11]. Cette dernière solution plonge ses racines dans la théorie des *Processus Décisionnels de Markov Partiellement Observables* (*Partially Observable Markov Decision Processes*, POMDP) [9], dans laquelle la perception de l'agent est une variable aléatoire de l'état courant. Néanmoins, ces approches antérieures sont sans rapport avec notre problème. Dans notre contexte, les ambiguïtés perceptuelles peuvent en effet toujours être éliminées, à condition d'affiner suffisamment le classificateur d'images. En fait, les techniques antérieures gèrent un *manque d'informations* inhérent aux capteurs utilisés, là où la nôtre gère un *surplus d'informations* produit par la haute redondance

des données visuelles.

3.4 Détecter les Classes Visuelles Ambiguës

Dans cette section, nous introduisons un nouveau critère permettant de tester la présence d'ambiguïté perceptuelle dans une classe [8].

Projection d'un MDP par un classificateur. Tout classificateur d'image $\mathcal{C}_k$ convertit une suite de N interactions $\langle p_t, a_t, r_{t+1}, p_{t+1}\rangle$ en une *suite projetée* de N quadruplets $\langle \mathcal{C}_k(p_t), a_t, r_{t+1}, \mathcal{C}_k(p_{t+1})\rangle$. Définissons le *MDP projeté* $\mathcal{M}_k$ comme étant le MDP $\langle S_k, A, \mathcal{T}_k, \mathcal{R}_k\rangle$ obtenu à partir de la suite projetée, où S_k correspond à l'ensemble des classes visuelles qui sont connues de $\mathcal{C}_k$, et où $\mathcal{T}_k$ et $\mathcal{R}_k$ sont estimées en calculant les fréquences relatives apparaissant dans la suite projetée.

Plus précisément, soient $i, j \in \{1, \ldots, m_k\}$ et soit $a \in A$. Définissons ensuite les fonctions suivantes :
- $\delta_t(V_{k,i}, a)$ vaut 1 quand $\mathcal{C}_k(p_t) = V_{k,i}$ et $a_t = a$, et 0 dans le cas contraire ;
- $\delta_t(V_{k,i}, a, V_{k,j})$ vaut 1 quand $\mathcal{C}_k(p_t) = V_{k,i}$, $a_t = a$ et $\mathcal{C}_k(p_{t+1}) = V_{k,j}$, et 0 dans le cas contraire ;
- $\eta(V_{k,i}, a)$ est le nombre de t pour lesquels $\delta_t(V_{k,i}, a)$ est égal à 1, c'est-à-dire $\eta(V_{k,i}, a) = \sum_{t=1}^{N} \delta_t(V_{k,i}, a)$.

Grâce à ces fonctions, nous pouvons définir formellement :
- $S_k = \{V_{k,1}, \ldots, V_{k,m_k}\}$;
- $\mathcal{T}_k(V_{k,i}, a, V_{k,j}) = \sum_{t=1}^{N} \delta_t(V_{k,i}, a, V_{k,j})/\eta(V_{k,i}, a)$;
- $\mathcal{R}_k(V_{k,i}, a) = \sum_{t=1}^{N} r_t \delta_t(V_{k,i}, a)/\eta(V_{k,i}, a)$.

Fonction de qualité optimale pour un MDP projeté. Par le théorème de Bellman (cf. section 2.2), nous savons que tout MDP projeté $\mathcal{M}_k$ possède une fonction de qualité optimale définie sur le domaine $S_k \times A$, qui sera dénotée Q'^*_k. Il a été prouvé que Q Learning converge vers Q'^*_k quand on lui fournit en entrée la suite projetée générée à travers $\mathcal{C}_k$ [22][2]. De même, il est immédiat de voir que toutes les méthodes de RL basées sur un modèle (cf. section 2.3) convergent vers Q'^*_k, à condition que le MDP projeté soit utilisé comme modèle.

À son tour, la fonction Q'^*_k induit une nouvelle fonction de qualité Q^*_k définie sur le domaine initial $S \times A$ en utilisant la relation $Q^*_k(s, a) = Q'^*_k(\mathcal{C}_k(p_s), a)$, où p_s est une perception quelconque pouvant être générée par s.[3] En l'absence d'ambiguïté perceptuelle, l'agent peut agir de manière optimale et Q^*_k doit nécessairement correspondre à Q^* en vertu du théorème de Bellman qui affirme l'unicité des fonctions de qualité optimales.

Par l'équation (4), la fonction $B_k = HQ^*_k - Q^*_k$ définie sur le domaine $S \times A$ est par conséquent une mesure de l'ambiguïté inhérente au classificateur d'images $\mathcal{C}_k$. Dans la terminologie de l'Apprentissage par Renforcement, B_k est appelé *résidu de Bellman* de la fonction de qualité Q^*_k. L'idée principale de RLVC consiste à affiner les classes visuelles correspondant aux états qui ont un résidu de Bellman non nul.

Mesure de l'ambiguïté dans un cadre déterministe. Malheureusement, la transformée H est inconnue en pratique, puisque l'agent n'a pas un accès direct à l'espace d'états du MDP qui modélise l'environnement. Nous proposons une approche différente. Supposons que la fonction de transition $\mathcal{T}$ de l'environnement est déterministe. Dans ce cas, le résidu de Bellman B_k de Q^*_k peut être réécrit comme suit :

$$B_k(s,a) = \mathcal{R}(s,a) + \gamma \max_{a' \in A} Q^*_k(\mathcal{T}(s,a), a') - Q^*_k(s,a)$$

$$= \mathcal{R}(s,a) + \gamma \max_{a' \in A} Q'^*_k\left(\mathcal{C}_k\left(p_{\mathcal{T}(s,a)}\right), a'\right) - Q'^*_k(\mathcal{C}_k(p_s), a) \quad (6)$$

Considérons une action $a \in A$ et deux états $s, s' \in S$ de l'environnement. Il est important de remarquer que l'équation (6) permet de calculer directement le résidu de Bellman B_k pour le couple (s, a) si l'on dispose d'une interaction $\langle p_t, a_t, r_{t+1}, s_{t+1}\rangle$ telle que p_t a été générée par s, que p_{t+1} a été générée par s' et que $a_t = a$. En effet, dans ce cas, $B_k(s, a)$ vaut :

$$\Delta_t = r_{t+1} + \gamma \max_{a' \in A} Q'^*_k(\mathcal{C}_k(p_{t+1}), a') - Q'^*_k(\mathcal{C}_k(p_t), a_t). \quad (7)$$

En accord avec cette discussion, les résidus Δ_t mesurent l'ambiguïté perceptuelle induite par $\mathcal{C}_k$.[4] Un Δ_t non nul indique la présence d'ambiguïté perceptuelle dans la classe visuelle $V_t = \mathcal{C}_k(p_t)$ par rapport à l'action a_t. Notre critère permettant de tester la présence d'ambiguïté perceptuelle consiste dès lors à calculer la fonction Q'^*_k, puis à parcourir à nouveau toutes les interactions $\langle p_t, a_t, r_{t+1}, p_{t+1}\rangle$ pour détecter celles qui engendrent un Δ_t non nul.

3.5 Extraction des Descripteurs Distinctifs

Une fois que de l'ambiguïté perceptuelle a été détectée dans une certaine classe $V \in S_k$ par rapport à une action a, il faut encore sélectionner un nouveau descripteur local qui permettra de lever cette ambiguïté. Cela équivaut à découvrir un descripteur local qui explique le mieux possible les variations dans l'ensemble des valeurs Δ_t correspondant à V et à a. Il s'agit en fait d'un problème de classification, pour lequel nous proposons une adaptation de la méthode avec laquelle les arbres de décision sont généralement construits en apprentissage automatique [13].

Soit T l'ensemble des instants t tels que $\mathcal{C}_k(p_t) = V$ et $a_t = a$. Chaque seuil $c \in \mathbb{R}$ induit une partition de l'en-

[2] Ce résultat est loin d'être trivial, puisque *a priori*, il se peut qu'aucun MDP ne puisse générer la suite projetée, car cette dernière ne vérifie pas nécessairement l'hypothèse de Markov. Dès lors, si un algorithme de RL est appliqué directement sur la suite projetée, il peut très bien ne pas converger vers Q'^*_k, voire même ne pas converger du tout.

[3] On a en effet supposé que l'espace perceptuel est totalement observable à la section 2.1.

[4] Il est intéressant de noter que les valeur Δ_t correspondent exactement aux mises à jour appliquées par l'algorithme du Q Learning [25].

semble $\{p_t \mid t \in T\}$ des perceptions visuelles correspondant à la classe visuelle V : les images p_t pour lesquelles Δ_t est supérieur à c et les autres. L'idéal serait d'identifier une caractéristique visuelle qui diviserait l'ensemble $\{p_t \mid t \in T\}$ de la même manière qu'un seuillage avec un certain réel c.

À cet effet, nous trions la suite des Δ_t tels que $t \in T$. Pour chaque point de rupture c dans cette suite triée, nous sélectionnons la caractéristique visuelle qui maximise le gain d'information apporté par la caractéristique sur la partition induite par c. Cela est fait en envisageant tous les descripteurs locaux qui décrivent le voisinage de tous les points d'intérêts des images présentes dans l'ensemble $\{p_t \mid t \in T\}$. Au final, nous affinons la classe V en utilisant le descripteur local qui a donné le meilleur gain d'information.

3.6 Environnement Non Déterministes

Nous avons supposé depuis la section 3.4 que la fonction de transition $\mathcal{T}$ est déterministe. Évidemment, ce n'est pas le cas en général. Par conséquent, avant d'affiner le classificateur, nous utilisons un test d'hypothèse χ^2 afin de décider si la subdivision induite par la caractéristique visuelle fraîchement sélectionnée est différente d'une subdivision aléatoire. Cette approche s'inspire des techniques d'élagage des arbres de décision [16].

3.7 Algorithme

En rassemblant tous les éléments des sections précédentes, RLVC peut se décrire sous la forme de l'algorithme suivant :

1. Construire un compteur $k := 0$ et un classificateur d'images $\mathcal{C}_0$ qui lie toutes les perceptions à la même classe visuelle.
2. Calculer une fonction de qualité optimale Q'^*_k pour le MDP projeté *via* $\mathcal{C}_k$, en utilisant un algorithme d'Apprentissage par Renforcement (e.g. Q Learning). Enregistrer dans une base de données toutes les interactions $\langle p_t, a_t, r_{t+1}, p_{t+1}\rangle$ qui ont été rencontrées durant ce processus de RL.
3. Poser $\mathcal{C}_{k+1} := \mathcal{C}_k$.
4. Pour chaque $(i, a) \in \{1, \ldots, m_k\} \times A$:
   (a) Sélectionner toutes les interactions $\langle p_t, a_t, r_{t+1}, p_{t+1}\rangle$ présentes dans la base de données telles que $\mathcal{C}_k(p_t) = V_{k,i}$ et que $a_t = a$.
   (b) Pour chacune d'elles, calculer le résidu suivant :
   $$\xi_t = r_{t+1} + \gamma \max_{a' \in A} Q'^*_k(\mathcal{C}_k(p_{t+1}), a').$$
   Remarquons que puisque $Q'^*_k(\mathcal{C}_k(p_t), a_t) = Q'^*_k(V_{k,i}, a)$ est une constance c pour toutes les interactions sélectionnées, ξ_t est simplement le Δ_t de l'équation (7), décalé par la valeur c.
   (c) Si l'un de ces ξ_t est différent de c, sélectionner un nouveau descripteur local qui explique le mieux possible les variations présentes dans l'ensemble des ξ_t. Si le test d'hypothèse χ^2 réussit, affiner le classificateur $\mathcal{C}_{k+1}$ en subdivisant la classe visuelle $V_{k,i}$ en fonction du descripteur fraîchement sélectionné.
5. Si $\mathcal{C}_k \neq \mathcal{C}_{k+1}$, faire $k := k + 1$ et recommencer à l'étape 2.

À la fin de l'algorithme, le classificateur $\mathcal{C}_k$ est supposé sans ambiguïté perceptuelle. Il peut alors être utilisé pour contrôler le système.

4 Expériences

Nous avons évalué RLVC sur une tâche abstraite qui reste très proche d'un scénario robotique réel [17, 21], tout en évitant toute complexité non nécessaire. RLVC a réussi à résoudre la tâche de navigation visuelle, continue et bruitée représentée à la figure 2. Le but de l'agent est d'atteindre le plus vite possible une des deux sorties du labyrinthe. L'ensemble des localisations possibles est continu. En chaque point du labyrinthe, l'agent a quatre actions possibles : aller en haut, à droite, en bas ou à gauche. Chaque mouvement est altéré par un bruit gaussien, dont l'écart type est de 2% la taille du labyrinthe. Des murs de verre sont présents dans le labyrinthe. Si un déplacement amène l'agent dans un mur ou en-dehors du labyrinthe, la position de l'agent n'est pas modifiée.

L'agent reçoit une récompense de 100 quand il atteint une des sorties. Tout autre déplacement, y compris ceux qui sont interdits, produit un renforcement nul. Quand l'agent réussit à s'échapper du labyrinthe, il atteint un état terminal hors duquel il ne peut sortir et dans lequel toute action donne un renforcement nul. Dans cette expérience, γ a été fixé à $0,9$, mais les résultats de l'algorithme sont indépendants de cette valeur. Remarquons que l'agent est confronté au problème de la récompense tardive (cf. section 2.2). Notons aussi que l'agent doit pouvoir à tout instant estimer sa distance aux deux sorties afin de résoudre cette tâche de manière optimale.

La surface du labyrinthe est tapissée avec une image de 1280×1280 pixels consistant en un assemblage d'images de la base de données COIL-100 [14]. L'agent n'a pas un accès direct à sa position (x, y) dans le labyrinthe. Au lieu de cela, ses capteurs retournent une photographie de la portion de sol au-dessus de laquelle il se situe. Les murs de verre sont transparents, si bien que les capteurs retournent aussi les portions de l'image situées derrière les murs. Par conséquent, l'agent n'a aucun moyen de localiser directement les murs : il est obligé de les identifier comme les régions du labyrinthe dans lesquelles une action donnée ne change pas sa position.

Dans cette expérience, nous avons utilisé les *invariants couleur différentiels* comme descripteurs locaux [6]. Ces

FIG. 2 – Une tâche de navigation continue et bruitée. Les sorties du labyrinthe sont indiquées par des carrés avec une croix. Les murs de verre sont identifiés par des lignes continues. L'agent est dessiné au centre de l'image. Chacun des quatre mouvements possibles est représenté par une flèche, dont la longueur correspond à la distance parcourue par l'agent. Les capteurs de l'agent retournent une image qui correspond à une photographie de la zone encadrée par des pointillés.

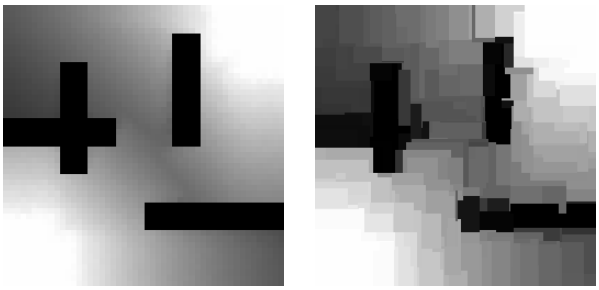
FIG. 3 – La liaison images-vers-actions déterministe obtenue, à savoir $\pi^* = \text{argmax}_{a \in A} Q_k^*(s,a)$. Celle-ci a été échantillonnée à des positions de l'agent espacées régulièrement. Cette liaison π^* réussit à choisir l'action correcte en chaque localisation.

descripteurs correspondent à des vecteurs de réels de dimension 8, donc $F = \mathbb{R}^8$. Les points d'intérêts sont détectés avec le détecteur *Harris couleur* [6]. L'entièreté de la surface du labyrinthe comprend 2298 différentes caractéristiques visuelles.

RLVC a sélectionné 200 caractéristiques, ce qui correspond à 9% du nombre total de caractéristiques disponibles. Le calcul s'est arrêté quand k a atteint la valeur 84 (soit après 35 minutes sur un Pentium IV cadencé à 2,4 GHz), ce qui correspond à peu près à la superficie du labyrinthe divisée par la longueur des déplacements de l'agent. 205 classes visuelles ont été identifiées. Il s'agit d'un petit nombre par rapport au nombre de classes perceptuelles qui seraient générées par un processus de discrétisation, i.e. si les perceptions de l'agent lui donnaient directement sa position (x, y) dans le labyrinthe sans passer par une image. Par exemple, si le labyrinthe était échantillonné sur une grille de 20×20 éléments, on obtiendrait 400 classes perceptuelles. La figure 3 montre la liaison images-vers-actions optimale qui résulte du dernier classificateur d'images $\mathcal{C}_k$ obtenu par RLVC.

En Apprentissage par Renforcement, il est généralement instructif d'étudier la *fonction de valeur optimale* $V^*(s)$. Cette fonction correspond au retour actualisé moyen que l'agent peut espérer obtenir en chaque état à condition d'agir de manière optimale, c'est-à-dire $V^*(s) = \max_{a \in A} Q^*(s,a)$ pour tout $s \in S$. La figure 4 compare la fonction de valeur optimale du problème discrétisé (i.e. quand l'agent a un accès direct à sa position (x, y) dans le labyrinthe) à celle obtenue *via* RLVC. La similarité entre ces deux fonctions indique l'adéquation de notre approche. Il est important d'insister sur le fait que RLVC fonctionne sans pré-traitement, ni intervention humaine. Initialement, l'agent ne sait pas quelles sont les caractéristiques visuelles qui sont pertinentes pour sa tâche. En outre, l'intérêt de sélectionner les descripteurs locaux discriminants pour construire un arbre de décision est évident dans cette application : si l'agent devait envisager toutes les combinaisons possibles de caractéristiques visuelles, le nombre de perceptions possibles s'élèverait à 2^{2298}.

Le comportement de RLVC a aussi été étudié sur des images du monde réel. Les règles de navigation ont été gardées identiques, mais l'image de fond a été remplacée par une photographie de 3041×384 pixels d'une station du métro parisien, comme montré à la figure 5 (a). L'algorithme a stoppé quand k a atteint la valeur 101. La liaison images-vers-actions qui a ainsi été calculée est montrée à la figure 5 (b). RLVC a sélectionné 144 caractéristiques visuelles parmi un ensemble de 3739 candidates, générant ainsi 149 classes visuelles. Au total, ce calcul a pris 159 minutes sur un Pentium IV cadencé à 2,4 GHz. Ici aussi, le classificateur d'images obtenu est suffisamment fin pour pouvoir obtenir une liaison images-vers-actions presqu'optimale. Certaines réactions le long des murs sont sub-obtimales, ce qui provient de la faible épaisseur des murs



(a) (b)

FIG. 4 – (a) La fonction de valeur optimale, quand l'agent a un accès direct à sa position (x, y) dans le labyrinthe et quand l'ensemble des localisations possibles est discrétisé dans une grille de taille 50×50. Plus une localisation est blanche, plus sa valeur est élevée. (b) La fonction de valeur optimale obtenue *via* le classificateur d'images obtenu par RLVC.

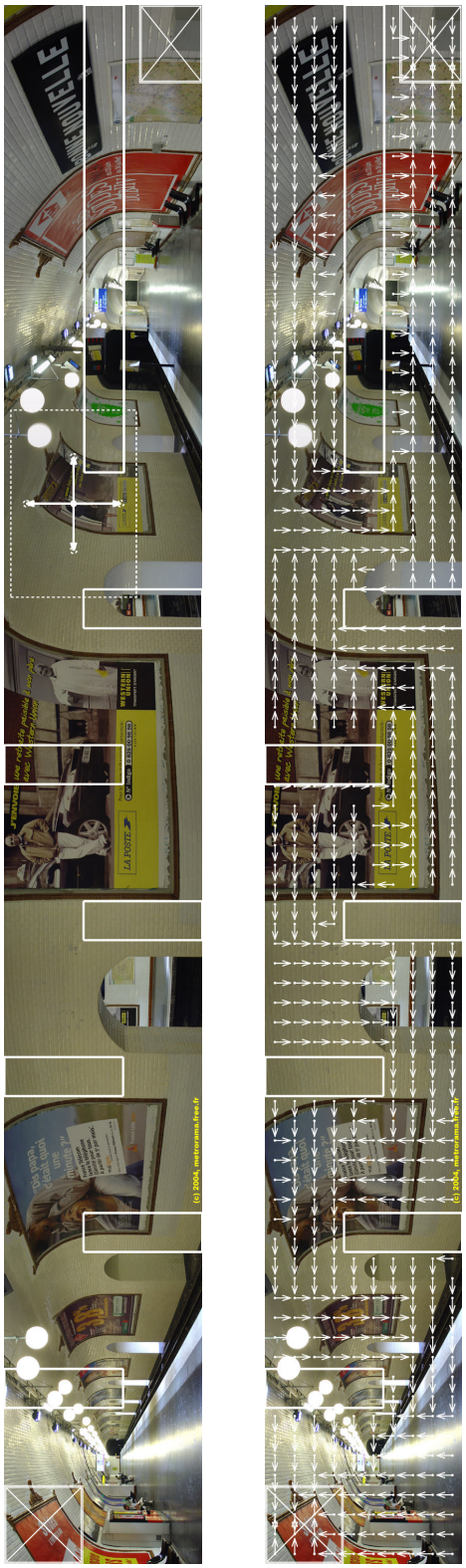
par rapport au champ de vision de l'agent. Notre approche semble dès lors applicable à des applications réalistes de vision par ordinateur.

5 Conclusions

Nous avons introduit un nouvel algorithme de vision par ordinateur, nommé *Apprentissage par Renforcement de Classes Visuelles* (*Reinforcement Learning of Visual Classes*, RLVC). RLVC a été conçu pour permettre l'apprentissage de liaisons directes de stimuli visuels vers des actions, de manière à maximiser une fonction de renforcement au cours du temps. Le processus d'apprentissage est très flexible. Il consiste à tirer des leçons sur la manière de se comporter dans un environnement *a priori* inconnu, à partir d'une suite d'interactions. Un tel apprentissage interactif est similaire à la manière dont les êtres vivants apprennent à résoudre leurs tâches de tous les jours.

RLVC concentre l'attention d'un algorithme enfoui d'Apprentissage par Renforcement sur des motifs hautement informatifs et robustes présents dans les stimuli visuels. À cet effet, un classificateur d'images teste la présence de descripteurs locaux aux points d'intérêts des images. Les caractéristiques visuelles pertinentes sont sélectionnées de manière incrémentale dans une suite de tentatives pour réduire l'ambiguïté perceptuelle. Elles sont progressivement arrangées dans une structure de données arborescente.

Il faut noter que le processus de sélection est guidé par la tâche : deux tâches différentes ne mèneront pas nécessairement à la sélection des mêmes caractéristiques. Cela est similaire au processus d'apprentissage visuel humain, pour lequel il existe des éléments qui suggèrent que de nouvelles classes visuelles sont apprises quand la tâche le demande [20]. De plus, l'addition incrémentale de nou-



(a) (b)

FIG. 5 – (a) Une tâche de navigation sur une image réelle, en utilisant les mêmes conventions que celles de la figure 2. (b) La liaison déterministe images-vers-actions calculée par RLVC pour cette tâche.

velles caractéristiques peut être vue comme un gain progressif d'expertise de l'agent sur sa tâche, tout comme dans la vision humaine [24]. En ce sens, le paradigme de RLVC peut être motivé du point de vue de la neuropsychologie, et non seulement comme un algorithme d'apprentissage automatique *ad hoc*. Notons finalement que RLVC n'est pas seulement applicable aux tâches visuelles, mais aussi à tous les domaines perceptuels sur lesquels il est possible de définir la notion de caractéristique binaire [8].

Le champ d'applications est large, puisque les agents robotiques sont de plus en plus souvent équipés de capteurs CCD. Notre but à long terme serait de construire un système robotique capable d'apprendre à structurer automatiquement ses perceptions visuelles afin de résoudre une tâche interactive, comme par exemple la saisie d'objets [15]. Une autre application possible serait d'utiliser RLVC dans le contexte de la classification d'images. En effet, la tâche de la classification d'images peut être formulée sous la forme d'un Processus Décisionnel de Markov, où les états seraient les images à classifier, où les actions consisteraient à attribuer une classe à l'état-image courant, et où la fonction de renforcement vaudrait 1 si la classe attribuée est correcte et 0 sinon. Dans cette dernière application, il n'y a pas d'aspect temporel, donc le coefficient d'actualisation γ vaudrait 0.

Toutefois, de nombreuses questions restent ouvertes. Par exemple, l'adaptation de RLVC à des espaces d'actions continus n'est pas directe. Or, ceci est nécessaire pour toute application robotique. Il serait également intéressant d'ajouter des opérations de *post-élagage* afin d'éliminer les descripteurs locaux qui se révèlent inutiles pour la tâche et qui génèrent des effets d'« *overfitting* ». Une autre voie de recherches prometteuse serait de combiner les caractéristiques visuelles de manière géométrique afin de produire des caractéristiques de haut niveau qui seraient à la fois plus discriminantes et plus robustes au bruit [15].

Références

[1] R. BELLMAN. *Dynamic Programming*. Princeton University Press, 1957.

[2] D.P. BERTSEKAS et J.N. TSITSIKLIS. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[3] L. CHRISMAN. « Reinforcement Learning with Perceptual Aliasing : The Perceptual Distinctions Approach ». Dans *National Conference on Artificial Intelligence*, pages 183–188, 1992.

[4] L. DEBNATH et P. MIKUSINSKI. *Introduction to Hilbert Spaces with Applications*. Academic Press, 1990.

[5] E.J. GIBSON et E.S. SPELKE. The Development of Perception. Dans John H. FLAVELL et Ellen M. MARKMAN, éditeurs, *Handbook of Child Psychology Vol. III : Cognitive Development*, Chapitre 1, pages 2–76. Wiley, 4th édition, 1983.

[6] V. GOUET et N. BOUJEMAA. « Object-based queries using color points of interest ». Dans *IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 30–36, Kauai (HI, USA), 2001.

[7] S. JODOGNE et J.H. PIATER. « Interactive Selection of Visual Features through Reinforcement Learning ». Dans M. BRAMER, F. COENEN et T. ALLEN, éditeurs, *Proc. of the 24th SGAI Internal Conference on Innovative Techniques and Applications of Artificial Intelligence*, volume 21 de *Research and Development in Intelligent Systems*, pages 285–298, Cambridge (Royaume-Uni), décembre 2004. Springer-Verlag.

[8] S. JODOGNE et J.H. PIATER. « Reinforcement Learning of Perceptual Classes using Q Learning Updates ». Dans M.H. HAMZA, éditeur, *Proc. of the 23rd IASTED International Multi-Conference on Artificial Intelligence and Applications*, pages 445–450, Innsbruck, Autriche, février 2005. Acta Press.

[9] L.P. KAELBLING, M.L. LITTMAN et A.R. CASSANDRA. « Planning and Acting in Partially Observable Stochastic Domains ». *Artificial Intelligence*, 101(1-2) :99–134, 1998.

[10] D.G. LOWE. « Object Recognition from Local Scale-Invariant Features ». Dans *International Conference on Computer Vision*, pages 1150–1157, Corfou, Grèce, septembre 1999.

[11] R.A. MCCALLUM. « *Reinforcement Learning with Selective Perception and Hidden State* ». PhD thesis, Université de Rochester, New York, 1996.

[12] K. MIKOLAJCZYK et C. SCHMID. « A Performance Evaluation of Local Descriptors ». Dans *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 257–263, Madison (WI, USA), juin 2003.

[13] T.M. MITCHELL. *Machine Learning*. McGraw Hill, 1997.

[14] S.A. NENE, S.K. NAYAR et H. MURASE. « Columbia Object Image Library (COIL-100) ». Rapport Technique CUCS-006-96, Columbia University, New York, février 1996.

[15] J.H. PIATER. « *Visual Feature Learning* ». PhD thesis, Université du Massachusetts, Computer Science Department, Amherst, MA, février 2001.

[16] J.R. QUINLAN. The Effect of Noise on Concept Learning. Dans *Machine Learning : An Artificial Intelligence Approach : Volume II*, pages 149–166. Kaufmann, Los Altos (CA, USA), 1986.

[17] A. REMAZEILLES, F. CHAUMETTE et P. GROS. « Robot Motion Control from a Visual Memory ». Dans *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2004)*, volume 4, pages 4695–4700, New Orleans (LA, USA), avril 2004.

[18] C. SCHMID et R. MOHR. « Local Greyvalue Invariants for Image Retrieval ». *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5) :530–535, 1997.

[19] C. SCHMID, R. MOHR et C. BAUCKHAGE. « Evaluation of Interest Point Detectors ». *International Journal of Computer Vision*, 37(2) :151–172, 2000.

[20] P.G. SCHYNS et L. RODET. « Categorization Creates Functional Features ». *Journal of Experimental Psychology : Learning, Memory and Cognition*, 23(3) :681–696, 1997.

[21] S. SE, D.G. LOWE et J. LITTLE. « Global Localization using Distinctive Visual Features ». Dans *International Conference on Intelligent Robots and Systems (IROS 2002)*, pages 226–231, Lausanne, 2002.

[22] S.P. SINGH, T. JAAKKOLA et M.I. JORDAN. « Reinforcement Learning with Soft State Aggregation ». Dans *Advances in Neural Information Processing Systems*, volume 7, pages 361–368. MIT Press, 1995.

[23] R.S. SUTTON et A.G. BARTO. *Reinforcement Learning, an Introduction*. MIT Press, 1998.

[24] M.J. TARR et Y.D. CHENG. « Learning to See Faces and Objects ». *Trends in Cognitive Sciences*, 7(1) :23–30, 2003.

[25] C.J.C.H. WATKINS et P. DAYAN. « $Q-$learning ». *Machine learning*, 8 :279–292, 1992.

[26] S.D. WHITEHEAD et D.H. BALLARD. « Learning to Perceive and Act by Trial and Error ». *Machine Learning*, 7 :45–83, 1991.