# Coopération d'algorithmes d'apprentissage par renforcement multiples

## *Cooperation of multiple reinforcement learning algorithms*

*par* **Benoît GIRARD**

Directeur de Recherche CNRS

Institut des Systèmes Intelligents et de Robotique, ISIR (UMR7222, CNRS - UPMC)

**Mehdi KHAMASSI**

Chargé de Recherche CNRS

Institut des Systèmes Intelligents et de Robotique, ISIR (UMR7222, CNRS - UPMC)

### Résumé

Développées initialement dans le cadre de l'intelligence artificielle, les méthodes d'apprentissage par renforcement sont des composantes essentielles des architectures de contrôle robotique adaptatives. Deux grands classes d'algorithmes ont été proposées : avec ou sans modèle interne du monde. La première est coûteuse en calculs mais est très adaptative, alors que la seconde est peu coûteuse mais lente à converger. La combinaison de ces différents algorithmes dans une même architecture de contrôle permet donc d'envisager tirer le meilleur parti des deux mondes. Nous présentons ici ces deux familles d'algorithmes, ainsi que les méthodes de combinaison qui ont été proposées et évaluées, tant en neurosciences qu'en robotique.

### Abstract

Initially developed in the field of artificial intelligence, reinforcement learning methods are an essential component of adaptive robotic control architectures. Two main classes of algorithms have been proposed: with and without internal models of the world. The first one has heavy computational costs but is very adaptive, while the second one is cheap but slow to converge. The combination of these algorithms within a single robotic architecture could possibly benefit from the advantages of each one. We present here these two families of algorithms, as well as the combination methods that have been proposed and tested in the neuroscience and robotics field.

**Mots-clés : apprentissage par renforcement, méthodes d'ensemble, neuro-inspiration, neuro-robotique**

***Keywords: Reinforcement learning, ensemble methods, neuro-inspiration, neuro-robotics***

Table des matières



## *Introduction*

*Les méthodes d'apprentissage par renforcement sont des composantes essentielles du développement de systèmes robotiques autonomes. Elles doivent en effet permettre à ces systèmes d'apprendre, par essais et erreurs, sans intervention additionnelle de leurs concepteurs, les actions qui doivent être effectuées, et celles qui doivent être évitées, pour la réalisation de leur mission.*

*Deux grands classes d'algorithmes ont été historiquement développées dans la littérature : celle fondée sur l'utilisation d'un modèle interne du monde, et en particulier des transitions entre états, et celle sans modèle interne. La première est grande consommatrice de ressources computationnelles (i.e. calculs nécessaires pour déduire l'action qui semble aboutir aux meilleures conséquences telles que prédites par le modèle interne), mais permet de réagir en quelques essais aux changements de l'environnement en réutilisant les connaissances précédemment apprises sur la structure de l'environnement grâce au modèle interne ; la seconde est extrêmement peu coûteuse (pas de modèle, dont pas d'estimation des conséquences de l'action), mais au prix d'une convergence lente de l'apprentissage et d'une très mauvaise adaptabilité au changement (i.e. des centaines d'essais sont nécessaires pour mettre à jour les valeurs associées aux actions suite à un changement de l'environnement). Il pourrait donc sembler logique de chercher à bénéficier des complémentarités de ces deux approches en les combinant. Pourtant, la coopération de systèmes d'apprentissage par renforcement multiples a, jusqu'ici, été peu explorée dans la littérature de l'apprentissage automatique.*

*La mise en avant des bonnes propriétés d'une telle approche s'est donc initialement développée dans le contexte de l'étude du comportement animal. En effet, la cohabitation de systèmes d'apprentissage multiples, et l'existence de substrats neuronaux distincts, ont été clairement démontrées en neurosciences. Plusieurs modèles computationnels ont été proposés pour rendre compte de la manière dont les animaux coordonnent leurs systèmes d'apprentissage multiples. Ces modèles constituent une source d'inspiration pour la conception de systèmes robotiques. Cette importation a principalement eu pour cadre la navigation, mais ne doit pas nécessairement s'y limiter. Enfin, les limites de ces méthodes, dont l'objectif scientifique est la simulation du comportement animal et non l'efficacité opérationnelle, sont parfaitement dépassables dans le cadre de l'ingénierie, en se défaisant des contraintes biologiques.*

# 1. Apprentissage par renforcement

La conception d'un agent artificiel ayant un degré d'autonomie avancé dans la réalisation de sa mission s'appuie sur la mise en oeuvre de capacités de raisonnement et de décision qui sont au coeur des travaux de l'intelligence artificielle depuis ses débuts. Cependant, comme il est impossible, dans des cas d'usage un tant soit peu complexes, que le concepteur puisse prévoir l'ensemble des cas pouvant se présenter, cette autonomie doit également pouvoir s'appuyer sur des capacités d'apprentissage lui permettant d'intégrer de nouvelles informations et d'acquérir de nouvelles capacités d'action.

L'apprentissage automatique (*machine learning*) distingue trois grandes classes d'algorithmes d'apprentissage :

- L'apprentissage non-supervisé, qui a pour but d'apprendre à identifier les régularités statistiques d'un flux de données en entrée, afin de mettre en évidence les éventuelles structures cachées ayant généré ce flux. Il peut être utilisé pour catégoriser ces entrées, mais aussi pour en fournir une description plus compacte, destinée à servir d'entrée à d'autres algorithmes d'apprentissage.
- L'apprentissage supervisé cherche à fournir les sorties attendues pour un flux de données en entrée, sur la base d'un signal d'erreur. Ce signal d'erreur indique pour chaque couple entrée-sortie l'écart qu'il convient de réduire entre la sortie générée et la sortie souhaitée.
- L'apprentissage par renforcement doit apprendre à associer des entrées et des sorties afin de maximiser, sur le long terme, la somme des signaux de récompense obtenus (signaux qui peuvent prendre des valeurs négatives et correspondre alors à des punitions). Contrairement à l'apprentissage supervisé, ce signal n'est pas fourni continûment, mais occasionnellement, et n'indique pas à proprement parler une erreur (un décalage entre une sortie souhaitée et une sortie générée) mais simplement un résultat (positif ou négatif) d'une série de sorties générées [26].

La formulation même du problème de l'apprentissage par renforcement correspond précisément au type de problèmes rencontrés en robotique autonome et adaptative : il concerne en effet l'apprentissage par essais et erreurs d'un agent plongé dans un monde qu'il ne connaît pas a priori et dont il reçoit des messages de récompense épars et peu informatifs. De fait, de très nombreux travaux de robotique, implantés sur des plateformes variées (bras, robots mobiles terrestres ou volants, humanoïdes, etc.) et résolvant des tâches tout aussi variées (locomotion, manipulation, etc.), ont utilisé les algorithmes d'apprentissage par renforcement issus de l'apprentissage automatique [23].

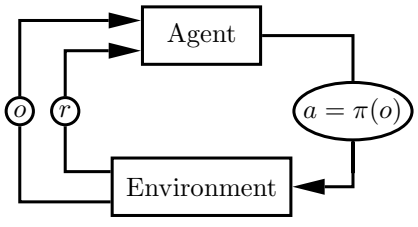


**Figure 1:** Paradigme de l'apprentissage par renforcement, o : observation ; r : récompense ; a : action.

La terminologie adoptée dans le cadre de l'apprentissage par renforcement est en général la suivante (Fig. 1) : les entrées sont des observations $o \in O$ de

l'environnement dans le lequel est plongé l'agent ; les sorties sont des actions $a \in A$ exécutées par l'agent; le signal de récompense *r* fourni occasionnellement par l'environnement doit être utilisé pour modifier les associations apprises entre les observations et les actions, associations couramment nommées « politique » ($a = \pi(o)$). Les observations et les actions peuvent être discrètes ou continues, et sont en général multi-dimensionnelles. L'objectif final est de trouver la politique maximisant la récompense accumulée sur le long terme, *G*. *G* peut être définie de multiples manières, mais l'une des formulations les plus utilisées est fondée sur un facteur d'atténuation $\gamma$ (équation 1).

(1) $$G(t) = \sum_{i=0}^{+\infty} \gamma^i r(t+i), \text{ avec } 0 < \gamma < 1 .$$

Ce paramètre permet d'ajuster le comportement souhaité : soit pousser le système à prendre en compte les récompenses dans un futur lointain, pour ajuster les choix courants, s'il est proche de 1, ou au contraire favoriser une optimisation myope, favorisant les retours à court terme, avec $\gamma$ proche de 0.

Cette définition de *G* permet en particulier de dériver simplement deux grandes familles d'algorithmes d'apprentissage : ceux où l'on va apprendre, maintenir et utiliser un modèle du monde pour prendre des décisions (algorithmes fondés sur un modèle), et ceux où l'on va se contenter d'apprendre par coeur les bonnes associations *(o,a)* sans modéliser la structure du monde sous-jacente (algorithmes sans modèle).

## 1.1 Algorithmes fondés sur un modèle

Les algorithmes d'apprentissage par renforcement avec modèles du monde vont chercher à découvrir incrémentalement, par exploration du monde, et à mémoriser la *fonction de transition T( o(t), a, o(t+1) ) = P( o(t+1) | o(t), a(t) )*, qui caractérise la probabilité de la prochaine observation *o(t+1)* sachant l'observation courante *o(t)* et l'action entreprise *a(t)*, et la *fonction de récompense*, qui caractérise les signaux de récompense reçus lorsque l'on exécute une action donnée face à une observation donnée *R(o(t),a(t))*. Sur la base de ces informations, il devient possible d'évaluer récursivement la valeur d'une observation *V(o)*, en fonction de la valeur des observations *o'* atteignables avec toutes les actions disponibles, par l'équation de Bellman (équation 2), où $\gamma$ est le facteur d'atténuation défini à l'équation 1.

(2) $$V(o) = \max_{a \in A}[R(o,a) + \gamma V(o')T(o,a,o'))]$$

On peut alors en déduire la politique correspondante (équation 3).

(3) $$\pi(o) = argmax_a[R(o,a) + \gamma V(o')T(o,a,o')]$$

Cette évaluation récursive est la base des algorithmes dits « d'itération de la valeur » (voir encadré, on notera l'utilisation d'une variable intermédiaire, *Q(o,a)*, évaluant la valeur de l'activation d'une action *a* pour une observation *o*).

**Nota** : Si le modèle du monde est connu à l'avance, ces algorithmes fondés sur un modèle se ramènent alors à des algorithmes de programmation dynamique classiques [2].

**Algorithme d'itération de la valeur**

Avant chaque prise de décision :
    initialiser V arbitrairement
    tant que la politique n'est pas assez bonne :
        pour tout o de O :
            pour tout a de A :
                $Q(o,a) \leftarrow R(o,a) + \gamma V(o')T(o,a,o')$
            fin pour
            $V(o) \leftarrow \max_a Q(o,a)$
            $\pi(o) \leftarrow argmax_a Q(o,a)$
        fin pour
    fin tant que

Plusieurs possibilités existent pour la définition d'un critère de fin, mais on peut se baser sur le résultat qui montre que si la différence entre deux valeurs successives de V(o) est inférieure à $\epsilon$, la politique ne diffère de la politique optimale que de $2\epsilon\gamma/(1-\gamma)$. Il est également possible d'itérer directement sur les politiques (on parle alors « d'itération de la politique »), et naturellement, ces algorithmes assez anciens ont connu beaucoup de raffinements afin d'en améliorer les performances [19]. Reste que globalement, ils reviennent à parcourir l'ensemble du graphe des transitions (que constitue le modèle) à de très nombreuses reprises afin de parvenir à une évaluation juste des valeurs des actions ou de la politique. Ils sont donc computationnellement très coûteux. Cela est illustré fig. 2 avec le problème-jouet proposé en A : un labyrinthe discret, dont chaque case produit une observation unique, où l'agent commence en S, reçoit une récompense de 1 lorsqu'il entre en R, et où il a quatre actions à disposition (se déplacer vers le Nord, l'Est, le Sud ou l'Ouest). Les valeurs d'état étant initialisées à 0, la première trajectoire menant au but ne peut être que le résultat d'une exploration aléatoire uniforme. Supposons que la réalisation de cette première trajectoire soit celle indiquée en bleu (fig. 2, A). A la suite de celle-ci, l'agent connait le graphe de transition (fig. 2, B), l'itération de valeur permet, d'étape en étape de propager l'information de valeur des états dans le graphe (ici, 8 étapes sont nécessaires, voir fig. 2, C, D et E).

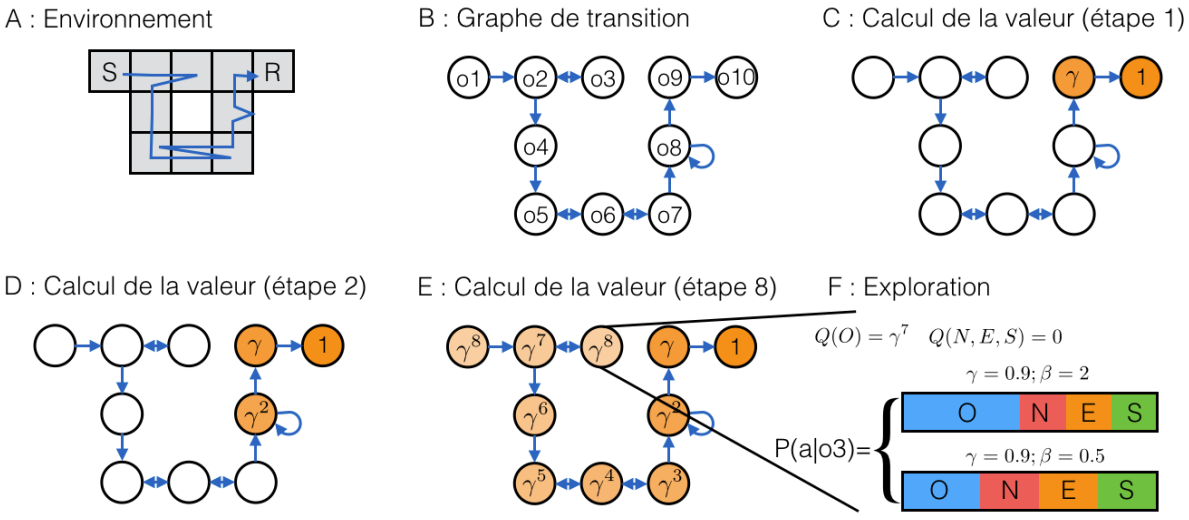

**Figure 2 :** Fonctionnement de l'algorithme d'iteration de la valeur. **A :** Problème-jouet de labyrinthe discret. **B :** graphe de transition déduit de la première trajectoire. **C-E :** propagation de l'estimation de valeur des états par l'algorithme d'itération de la valeur. **F :** Probabilités de choix des actions dans l'état o3, selon la valeur de beta.

Les algorithmes avec modèle interne nécessitent d'explorer l'environnement extensivement, afin d'acquérir une bonne estimation des fonctions de transition et de récompense. La conception d'une politique d'exploration efficace de l'espace des couples *(o,a)* est alors un problème en soi. En revanche, une fois cette exploration réalisée et si l'environnement est statique, une seule exécution de l'algorithme de calcul de la politique sera suffisante pour mettre en oeuvre la politique optimale pour toute observation. Si, après acquisition, la tâche change (la récompense n'est plus associée aux mêmes couples *(o,a)*, certaines transitions disparaissent ou apparaissent), il faudra être capable de l'identifier (par exemple en constatant un effondrement du taux de récompense moyen sur une fenêtre de temps récente). Il sera alors nécessaire d'effectuer une nouvelle phase d'exploration, pour remettre à jour le modèle du monde, et de refaire le calcul de la politique optimale.

Une solution alternative, mais coûteuse, consiste à lancer le calcul de la politique optimale à chaque pas de temps sans phase d'exploration préalable, en utilisant à chaque fois le modèle du monde déjà exploré. Ce modèle, initialement très pauvre, va s'enrichir au fur et à mesure de l'exploration. Afin d'assurer cette exploration, plutôt que de choisir systématiquement l'action supposée la meilleure ($\pi(o) \leftarrow argmax_a Q(o,a)$), choix qui sera de toute façon très mal informé en début d'apprentissage, on va tirer cette action suivant une distribution de probabilité. Un choix classique est de la calculer par une opération *softmax* (équation 4).

$$P(a|o) = \frac{\exp(\beta Q(o,a))}{\sum_{k \in A} \exp(\beta Q(o,k))} \quad (4)$$

Le paramètre d'exploration $\beta$ prend des valeurs positives. Lorsqu'il est inférieur à 1, l'équation 4 favorise l'exploration en réduisant le contraste entre les valeurs *Q(o,a)* des actions. A l'inverse, lorsque $\beta$ est supérieur à 1, l'équation 4 accentue ce contraste et pousse donc à l'exploitation des informations déjà acquises. Dans l'exemple de la figure 2, F, l'agent n'ayant jamais tenté l'action « Est », il ne peut pas encore savoir qu'il s'agit du meilleur choix, et compte tenu de sa connaissance actuelle, il estime la valeur *Q(O)* de l'action Ouest supérieure aux trois autres. Dans ce cas, la probabilité de choix de chacune des actions est très différente pour $\beta$=0.5 (choix de Q(O) à 30%) et $\beta$=2 (choix de Q(O) à 46%). A ce stade préliminaire de découverte de la tâche, il est plus intéressant de donner leur chance aux autres actions, plutôt que de se concentrer sur la seule qui ait été testée. Il est donc assez naturel de chercher à faire varier dynamiquement le paramètre $\beta$, en le faisant croître au fur et à mesure de l'apprentissage, et le baissant à nouveau lorsqu'un changement de tâche est détecté. Cela relève du méta-apprentissage et sort du cadre de cet article.

**Nota** : L'un des intérêts d'utiliser un *softmax* est de permettre d'exhiber le comportement de « correspondance de la probabilité » observé chez l'animal, à savoir que la probabilité du choix d'une action correspond à la probabilité qu'elle produise une récompense.

## 1.2 Algorithmes sans modèle

Le coût computationnel relativement prohibitif des algorithmes avec modèle interne, d'autant plus lorsque l'on manipule de grands espaces d'observations O et d'actions A, a amené le développement d'algorithmes sans modèle interne (aussi appelés algorithmes d'apprentissage par différence temporelle, TD pour l'anglais *temporal difference*). Dans ces algorithmes, le parcours du graphe des transitions (qui est une forme d'exploration simulée) est remplacé par une expérimentation directe dans l'environnement : chaque passage d'une observation à une autre, suite à une action donnée, sera utilisé afin de raffiner l'estimation de la valeur courante de cette action.

Le coeur de ces algorithmes est issu de l'observation que la récompense accumulée sur le long terme, G, que l'on cherche à maximiser, peut s'exprimer d'un pas de temps à l'autre de manière récursive (équation 5).

(5) $$G(t) = r(t) + \gamma G(t+1)$$

V cherchant précisément à évaluer, pour toute observation, la valeur de G, on peut alors écrire que pour un système dont l'apprentissage aurait convergé, l'équation 6 devrait être vérifiée (qui peut être reformulée en équation 7).

(6) $$V(o(t)) = r(t) + \gamma V(o(t+1))$$

(7) $$0 = r(t) + \gamma V(o(t+1)) - V(o(t))$$

Pour un algorithme n'ayant pas encore convergé, cette « différence temporelle » entre valeur estimée aux temps *t* et *t+1* (terme de droite de l'équation 7), habituellement nommée $\delta$, va prendre des valeurs positives ou négatives. Si elle est positive, cela signifie que l'action qui a été exécutée au temps *t* a plus de valeur que prévu (soit que l'on ait obtenu une récompense *r(t)* plus grande que prévue, soit que la valeur de l'état atteint *V(o(t+1))* soit plus importante que prévu). La valeur prédite de *o(t)* doit donc être augmentée, et la probabilité de choisir cette action lorsqu'on observe *o(t)* (qui dérive de la valeur de l'action dans cet état *Q(o,a)*) doit être augmentée également. Inversement, si $\delta$ est négative, la valeur de l'état atteint à *t+1* est moins grande que prévue, et il faut donc diminuer *V(o(t))* et la probabilité de choisir *a* lorsqu'on observe *o(t)*. Ces variations sont donc proportionnelles à $\delta$ : le gain $\alpha$, compris entre 0 et 1, permet de contrôler la vitesse d'apprentissage. S'il est trop proche de 0 il peut générer un apprentissage très lent, mais trop proche de 1, il risque de causer des instabilités comportementales, si la réponse de l'environnement aux actions est stochastique. Cela peut se formuler sous la forme de l'algorithme acteur/critique (voir encadré), mais d'autres variantes similaires existent (algorithmes *Q-learning* et *SARSA* [26]). Plusieurs stratégies d'initialisation de V et Q peuvent être mise en oeuvre : 0 partout, *a priori* aucune récompense n'est attendue nulle part ; valeurs aléatoires faibles, qui vont donc générer une première politique *a priori* ; valeurs uniformes non-nulles, pour favoriser l'exploration initiale (les actions non testées sont *a priori* intéressantes).

**Algorithme Acteur-Critique**

Après chaque prise de décision à t+1:

$$\delta(t) \leftarrow r(t) + \gamma V(o(t+1)) - V(o(t))$$
$$V(o(t)) \leftarrow V(o(t)) + \alpha\delta(t)$$
$$Q(o(t), a(t)) \leftarrow Q(o(t), a(t)) + \alpha\delta(t)$$

La sélection de l'action à *t+1* s'effectue alors sur la base des *Q(o(t+1),a)*, par exemple en effectuant un tirage dans la distribution résultant d'un *softmax*, de la même manière que pour les algorithmes à modèle interne.

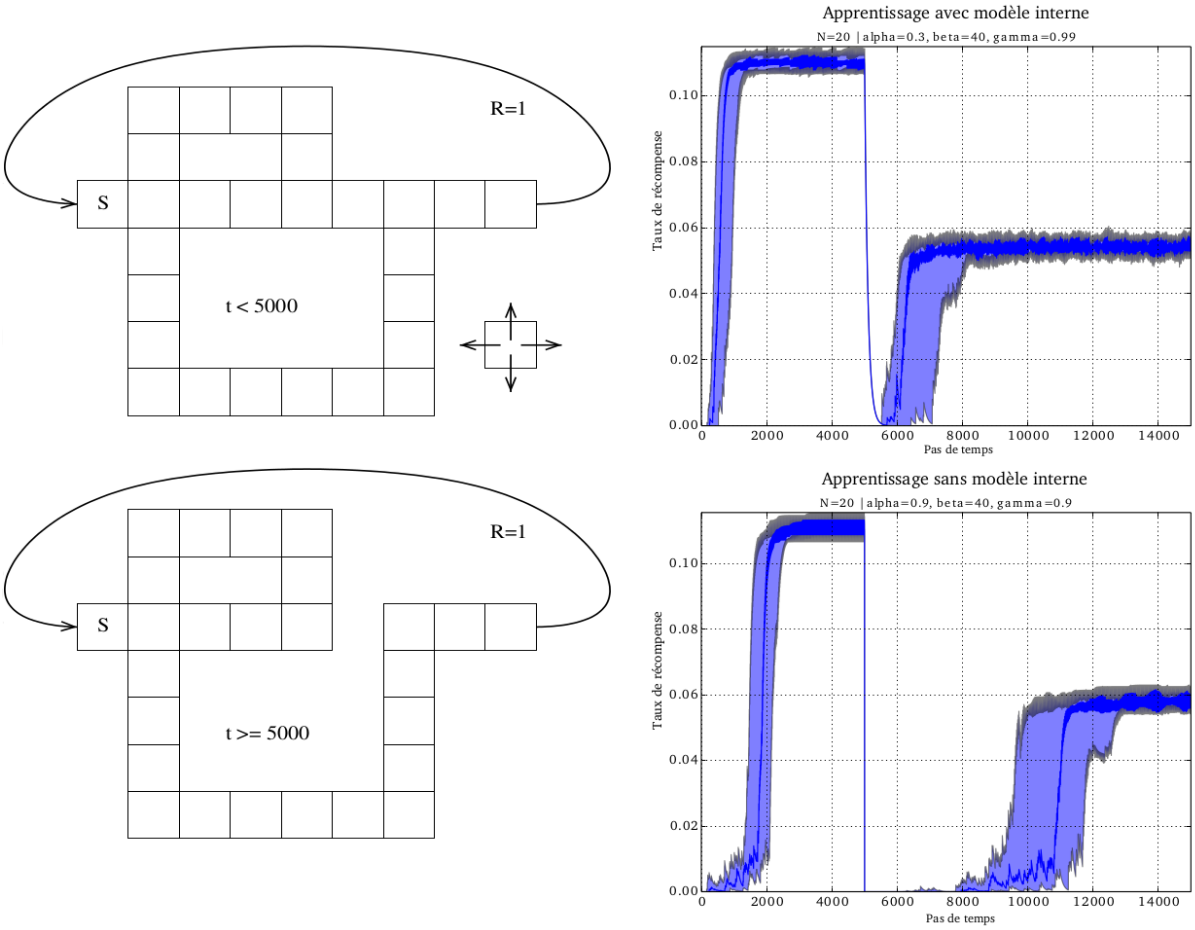


**Figure 3 : Gauche :** Tâche de test des algorithmes d'apprentissage par renforcement. **Droite :** résultats de simulations d'algorithmes d'apprentissage avec et sans modèle interne.

Des raffinements existent pour accélérer la convergence des algorithmes TD, par exemple en utilisant des traces d'éligibilité, qui mémorisent les observations rencontrées dans le passé, et permettent alors des mises à jour de valeur pour l'ensemble de ces observations plutôt que pour la dernière uniquement. Pour autant, ces algorithmes sont intrinsèquement plus lents à converger que les algorithmes avec modèle interne, il leur faudra effectuer un grand nombre d'essais et erreurs, pour chacune des observations possibles dans l'environnement, avant d'atteindre la politique optimale. Ils ont en revanche le grand avantage de ne demander que très peu de calculs à chaque itération (quelques sommes et multiplications, là où les algorithmes à modèle interne parcourent l'ensemble des combinaisons *(a,o)*). Il est à noter également que lorsque la tâche change (par exemple lorsque les couples *(a,o)* générant des récompenses changent, même si les règles de transition ne changent

pas), ils sont encore plus lents à converger à nouveau que lors de l'apprentissage initial : il leur faudra en effet désapprendre progressivement la valeur des couples *(a,o)*, mais tant que ce désapprentissage ne sera pas complet, ils seront poussés à appliquer l'ancienne politique. Dans un même cas, un algorithme avec modèle interne, sitôt qu'il a identifié que la récompense n'est plus à l'emplacement initial, met à jour l'ensemble de ses valeurs et n'est plus poussé à appliquer l'ancienne politique. Ce cas de figure est illustré dans le cas d'école d'un labyrinthe discret (fig. 3, gauche) : l'agent commence en S, peut choisir une action de déplacement parmi quatre (Nord, Sud, Est, Ouest, si la case cible n'existe pas, l'agent reste dans l'état courant) ; dans la dernière case à droite, l'action Est ramène l'agent en S et rapporte une récompense de 1 ; après 5000 pas de temps dans la configuration du haut, une case disparaît (configuration du bas), l'agent doit apprendre une nouvelle politique. La figure 3, droite, illustre les performances de 20 simulations d'algorithmes avec (en haut) et sans (en bas) modèle interne. La courbe représente le taux de récompense médian (les surfaces bleues représentent l'intervalle entre le premier et le troisième quartile, les différences entre essais sont causées par l'exploration et peuvent être modifiées en jouant sur le paramètre $\beta$). Les deux algorithmes parviennent à converger initialement vers le taux optimal de récompense (0,111 récompenses par pas de temps). Lorsque la tâche change, l'algorithme avec modèle interne converge aussi vite que la première fois vers le nouvel optimal (0,059 récompenses par pas de temps), alors que l'algorithme sans modèle est bien plus lent à réapprendre. Cette meilleure performance de l'algorithme avec modèle interne lors d'un changement est obtenue au prix de calculs plus conséquents.

## 1.3 Combinaison d'algorithmes multiples

L'existence de ces deux grandes familles d'algorithmes, aux caractéristiques complémentaires, aurait pu suggérer la conception d'architectures les combinant, et exploitant donc au mieux ces complémentarités. On peut ainsi imaginer un agent initialement contrôlé par un système avec modèle interne, coûteux en calcul mais plus efficace, qui pourrait progressivement donner la main à un système sans modèle interne une fois que la tâche est bien connue, afin d'économiser des ressources de calcul. La littérature de l'apprentissage par renforcement a pourtant peu exploré cette piste [28], en particulier pour la gestion d'environnements non-stationnaires, où l'on pourrait assez naturellement alterner entre systèmes, selon que l'on détecte un changement de tâche ou pas.

C'est du côté des neurosciences computationnelles que de tels modèles ont été proposés. En effet, dans le cadre de l'étude de l'apprentissage de comportements conditionnés, les expérimentateurs ont mis en évidence deux modes principaux de comportement : les comportements dirigés vers un but et les comportements habituels. Les comportements dirigés vers un but sont caractérisés par le fait qu'ils sont gouvernés par les estimations des conséquences des actions, et sont plutôt exprimés au début de l'apprentissage d'une nouvelle tâche ; alors que les comportements habituels sont gouvernés par des associations stimulus-réponse relativement inflexibles, et qu'ils tendent à apparaître après l'exposition prolongée à une tâche stable [9,10]. Les comportements dirigés vers un but se caractérisent en particulier, du point de vue expérimental, par le fait qu'ils sont sensibles à la dévaluation : si un rat arrête d'appuyer sur un levier lorsque celui-ci ne provoque plus l'obtention d'une boulette de nourriture, c'est que ce rat exprime bien un comportement dirigé vers un but. Si ce même rat continue à appuyer alors qu'il n'en retire plus de bénéfice, c'est qu'il est passé en mode habituel, ce qui apparaît après

un long entraînement de l'animal dans un même environnement stable. Les comportements dirigés vers un but sont supposés impliquer des processus mentaux coûteux, qui ralentissent l'exécution de la tâche, alors que les processus habituels sont très rapides et ne demandent pas d'efforts cognitifs [1]. Ces similarités avec les propriétés des modèles d'apprentissage par renforcement ont abouti à la proposition théorique suivante : les comportements dirigés vers un but seraient contrôlés par un système d'apprentissage par renforcement avec modèle interne, alors que les comportements habituels seraient contrôlés par un système d'apprentissage par renforcement sans modèle interne [6]. Cette conception modulaire des processus d'apprentissage par renforcement s'accorde assez bien avec le fait que les substrats neuronaux de ces deux types de comportement sont distincts [22,21]. Plusieurs modèles computationnels, que nous allons présenter plus en détails ci-dessous, se sont donc intéressés à comprendre quels critères permettent d'expliquer les passages de l'un à l'autre de ces modes de comportements, selon les circonstances expérimentales. Exprimés dans le formalisme de l'apprentissage par renforcement, ils sont directement exploitables en dehors du champ des neurosciences, et en particulier en robotique, d'autant plus qu'ils ont été dans un grand nombre de cas appliqués à des tâches de navigation, c'est à dire à l'une des fonctions essentielles d'un robot autonome. Nous les mettrons en perspective avec les quelques travaux issus directement de la robotique qui mettent en application de telles architectures d'apprentissage modulaires.

# 2. Méthodes de coordination d'algorithmes d'apprentissage

Globalement, trois grandes familles de combinaisons d'algorithmes d'apprentissage multiples ont été proposées : la *fusion* des sorties de ces algorithmes avant de prendre une décision, la *sélection* d'un de ces algorithmes qui prend alors seul le contrôle de l'agent, cette sélection pouvant résulter du *suivi de l'évolution de variables internes* ou encore d'une deuxième couche d'*apprentissage*.

## 2.1 Fusion statique

Si l'on ne cherche pas à optimiser l'utilisation des ressources de calcul, mais uniquement à améliorer le comportement d'un agent, on peut systématiquement calculer les sorties de l'ensemble des systèmes d'apprentissage, et ensuite les fusionner avant de prendre une décision. L'idée est alors que les actions qui font consensus sont probablement les meilleures.

La première méthode de coordination d'algorithmes d'apprentissage [13] a été proposée pour expliquer l'apprentissage de rats dans des tâches de navigation, avant et après lésion d'une partie du cerveau appelée l'hippocampe et connue pour être impliquée dans la localisation spatiale. Cette méthode combinait deux algorithmes d'apprentissage par renforcement sans modèle interne, utilisant des données d'entrée différentes : d'un côté, la configuration locale du labyrinthe (couloirs, intersections, culs de sac, etc.), de l'autre, une estimation de la localisation spatiale dans le référentiel du labyrinthe. L'hippocampe étant supposé être la région du cerveau en charge de l'évaluation de la localisation, sa lésion correspondrait à la désactivation de l'algorithme correspondant dans cette méthode. La combinaison des deux algorithmes était simplement réalisée en sommant les valeurs $Q(o,a)$ associées à chaque direction possible de déplacement, avant d'effectuer la sélection

de l'action. C'est une méthode tout à fait similaire qui a été utilisée par [14,15] pour cordonner un algorithme d'apprentissage sans modèle interne avec un algorithme d'apprentissage avec modèle interne, dans une tâche de navigation simulée. Les complémentarités des deux algorithmes ont été mises en évidence en comparant les performances de la combinaison avec celles des algorithmes isolés.

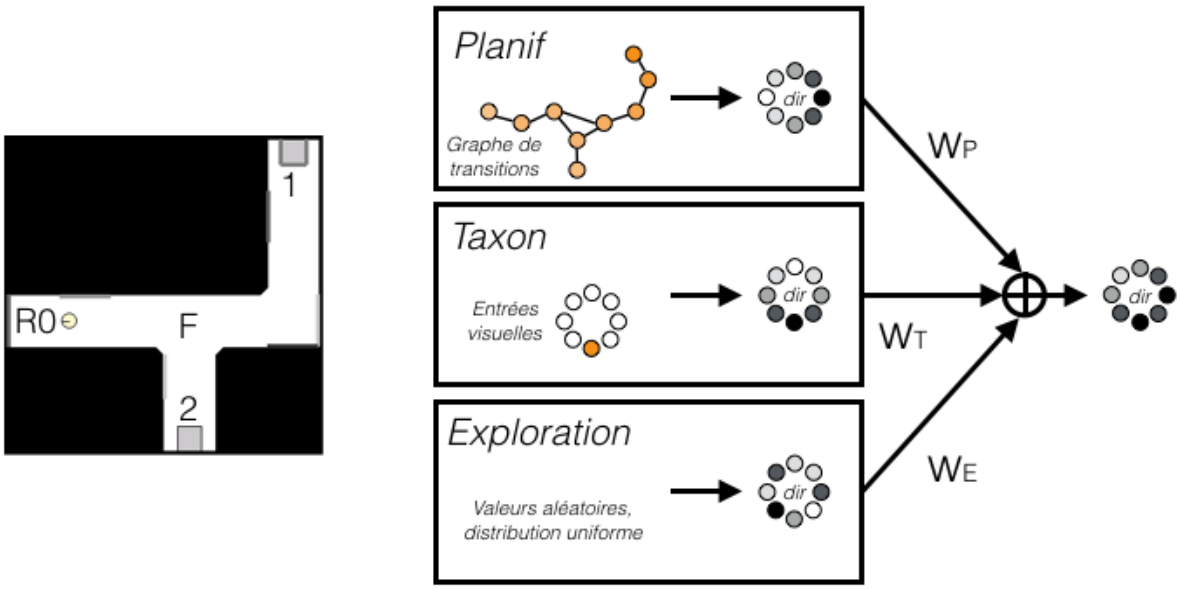


**Figure 4 : Gauche :** Tâches de test de conflit entre stratégies. **Droite :** Fusion par somme pondérée des valeurs des actions, illustrée pour la position F, utilisée dans Girard et al. [12].

Une idée similaire a été utilisée dans la méthode de [12], destinée à la robotique autonome : un algorithme d'apprentissage avec modèle interne (fondé sur la localisation), y était combiné avec un algorithme d'approche visuelle (équivalent au comportement après convergence d'un algorithme d'apprentissage sans modèle interne, travaillant sur les entrées visuelles, cadre « taxon » sur la fig. 4, droite). Les valeurs associées aux directions de déplacement étaient là aussi sommées, mais également pondérées (poids $w_P$ et $w_T$ sur la fig. 4), afin de pouvoir favoriser l'un ou l'autre des algorithmes en cas de conflit. Une première série de tests a permis de montrer que la combinaison de ces algorithmes était plus efficace que sans l'apprentissage avec modèle interne. Les tests suivants ont également montré que la pondération des valeurs permettait effectivement de favoriser la carte ou la vision dans la résolution d'un conflit, selon la pondération réglée par le concepteur. L'un de ces tests est illustré par la figure 4: le robot a déjà exploré l'environnement où seule la source de récompense 1 était présente, c'est donc la seule répertoriée par l'algorithme d'apprentissage avec modèle interne (« planif » sur la fig. 4, droite), lors du test, il est placé initialement en R0. Lorsqu'il arrive à l'intersection F, les deux algorithmes sont en conflit (fig. 4, droite) : la planification dans la carte recommande d'aller vers l'Est, l'approche visuelle, vers le Sud. Selon les valeurs de $w_P$ et $w_T$, le robot choisira préférentiellement d'une ou l'autre solution : fig. 4, droite, les poids $w_P$ et $w_T$ sont identiques et le choix résultera donc du hasard introduit par le module d'exploration.
Naturellement, la capacité d'ajuster ces poids automatiquement, en fonction de l'expérience, renforcerait l'adaptabilité du dispositif.

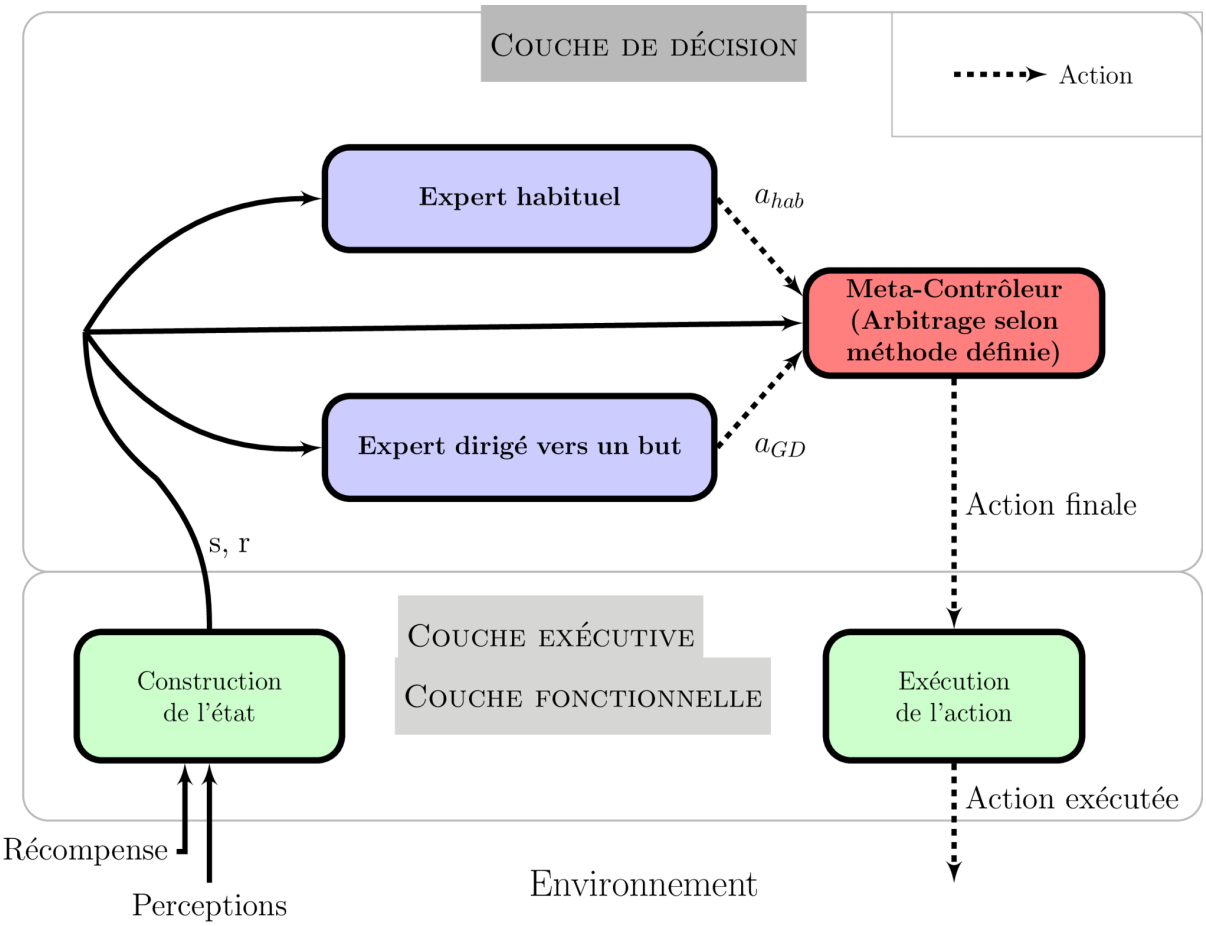


**Figure 5:** Architecture de [25] : les algorithmes d'apprentissage (experts) et le système de fusion (meta-contrôleur) sont insérés dans une architecture robotique classique à trois couches.

Enfin, Renaudo et al. [25] ont testé dans une tâche robotique simple (poussée de blocs sur un tapis roulant) la combinaison d'algorithmes avec et sans modèle interne, dans le cadre d'une architecture classique à 3 couches (fig. 5). Une batterie de critères de fusion, en partie issus des propositions de Wiering & van Hassel [28], ont été comparés à la performance d'un contrôle de complexité minimale (issue de la sélection aléatoire de l'un des deux algorithmes à chaque décision), face à un changement de tâche (changement de la vitesse du tapis). Les méthodes de fusion ne se sont pas toutes avérées capables de faire mieux que le contrôle ; la somme non pondérée des valeurs (similaire à la méthode de Guazzelli et al. [13]) ainsi que le vote ordonné, ont donné les meilleures performances.

**Nota** : Le vote ordonné fonctionne comme suit : pour chaque algorithme, les actions sont ordonnées suivant leur valeur estimée, on somme ensuite ces rangs et on normalise pour calculer les probabilités d'action.

Ces quelques résultats tendent à confirmer expérimentalement que la combinaison de valeurs d'actions estimées par des algorithmes différents, permet de bénéficier de leurs complémentarités et ainsi d'améliorer le comportement global de l'agent.

## 2.2 Sélection par suivi dynamique de variables internes

Les méthodes de fusion statique, si elles sont capables d'améliorer les performances globales d'un agent, ne permettent ni d'exclure de la décision un algorithme qui fonctionne mal à un instant donné, ni d'économiser des ressources de calcul en

désactivant un algorithme lorsqu'il n'est pas utile pour exhiber une bonne performance. Ces deux points sont pourtant au coeur des complémentarités potentielles entre algorithmes. Lors d'un changement de tâche, un algorithme avec modèle interne va réapprendre plus vite (voir fig. 3 en haut à droite), le fusionner avec un algorithme sans modèle interne (moins performant) est susceptible de dégrader le comportement de l'agent. A contrario, lorsqu'une tâche stable est bien apprise, il est inutile de continuer à effectuer les calculs complexes d'un algorithme avec modèle interne, alors que celui sans modèle interne est aussi performant.

Plusieurs méthodes ont donc été proposées pour effectuer la sélection d'un algorithme, qui prend alors seul le contrôle de l'agent pour la prochaine décision. La première méthode de cette catégorie [6] est issue de travaux de neurosciences computationnelles et cherchait à expliquer le passage de comportements dirigés vers un but à des comportements habituels chez des rats dans une tâche d'apprentissage conditionné (face à deux leviers, le rat doit apprendre lequel des deux entraînera la livraison de nourriture dans sa mangeoire). Cette méthode combine des variantes bayésiennes d'algorithmes d'apprentissage par renforcement avec et sans modèle interne, ce qui permet d'évaluer explicitement au cours de la progression de l'apprentissage leur niveau d'incertitude sur la connaissance de la tâche. L'incertitude est utilisée pour sélectionner à chaque instant l'algorithme le moins incertain : celui avec modèle interne en début de tâche, celui sans modèle interne ensuite. Le calcul explicite de l'incertitude est assez complexe et nécessite que les calculs de chaque algorithme soient effectués, de sorte qu'il n'y a pas au final de possibilité d'effectuer des économies de calcul. Il faut enfin noter que le choix de l'algorithme sans modèle interne en fin de tâche est dû à un handicap systématique attribué au système avec modèle interne, ce qui est tout à fait discutable.

La méthode proposée par Keramati et al. [20] est une modification de celle de Daw et al. [6] permettant d'y intégrer la notion de coût computationnel : l'algorithme sans modèle interne est systématiquement calculé, et son incertitude comparée au coût qu'il y aurait à effectuer les calculs de l'algorithme avec modèle interne (un paramètre de gain, fixé arbitrairement, est utilisé pour comparer l'incertitude et le coût). Ce n'est que lorsque l'incertitude de l'algorithme sans modèle interne est forte, et donc que le gain potentiel d'information via l'utilisation de l'algorithme avec modèle interne est supérieur à son coût, que ce dernier sera effectivement calculé. Cette méthode issue des neurosciences computationnelles s'est aussi intéressée à modéliser le comportement de rats en apprentissage conditionné, mais cherche à expliquer au delà des seuls choix effectués par les animaux, leurs temps de réponses, supposés liés aux efforts mentaux fournis. Cette méthode a pour défaut principal de fonder sa sélection d'algorithme sur l'hypothèse que l'algorithme avec modèle interne a une connaissance parfaite de l'environnement (fonctions de transition et de récompense), afin de pouvoir calculer la valeur de l'information parfaite (VPI pour « *value of perfect information »*). Cette connaissance parfaite est rarement vérifiée en pratique, à tout le moins durant les phases d'exploration.

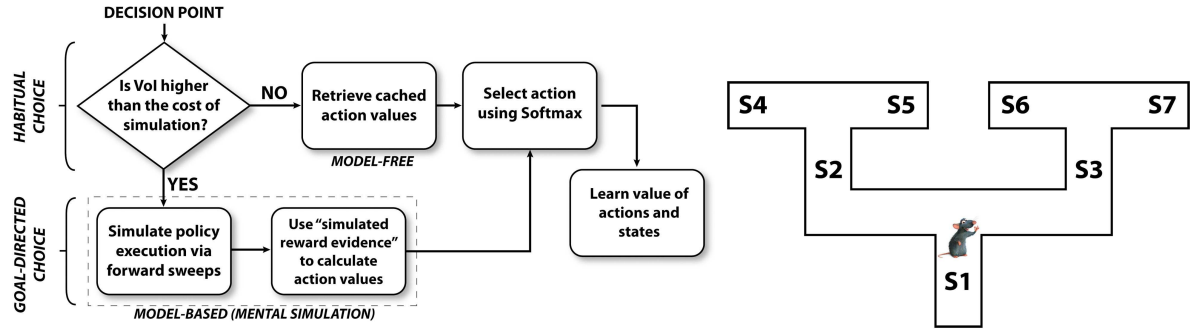


**Figure 6 : Gauche :** méthode de sélection proposée par [24]. **Droite :** labyrinthe en « double T » simulé dans les tests.

Ce dernier point a été corrigé dans la méthode de [24] (Fig. 6, gauche) : le calcul du gain d'information (VoI pour « *value of information »*) ne nécessite plus de supposer que l'algorithme avec modèle interne est parfait. Par ailleurs, l'incertitude n'est plus évaluée analytiquement, mais empiriquement par le biais d'un filtre à particules. Enfin, la quantité de ressources utilisées s'adapte plus finement, puisque la profondeur de l'exploration du graphe de transitions s'adapte également en fonction de l'incertitude sur la valeur de l'action. Cette méthode a été testée dans des tâches de navigation simulées, où l'agent doit prendre des décisions dans un labyrinthe en « double T » (Fig. 6, droite, S1 à S7 sont les observations possibles de l'agent dans cette tâche). Mais elle n'a pour l'instant été ni comparée directement à des données animales, ni évaluée sur une plateforme robotique.

Dans le cadre de la robotique, [16] puis [18] ont proposé de conditionner le changement de l'algorithme contrôlant actuellement le comportement du robot au dépassement d'un seuil de *frustration*. Dans le premier cas, les algorithmes entre lesquels il faut arbitrer sont un algorithme d'intégration de chemin et un algorithme d'apprentissage sans modèle interne. La frustration est alors définie comme dérivant de l'évolution d'une estimation de la distance au but : si cette distance stagne ou régresse trop longtemps, la frustration passe un seuil de déclenchement qui change alors l'algorithme en charge de contrôler le robot. Cette méthode ne fait cependant sens que dans le cas d'une tâche de navigation, et suppose que l'on est capable d'estimer la distance au but, ce qui n'est pas nécessairement le cas. La seconde méthode proposée est plus générale : elle cherche à prédire en continu les flux sensorimoteurs expérimentés par l'agent (un vecteur contenant toutes les informations sensorielles et motrices à un instant donné), les erreurs sur ces prédictions constituant une estimation de la nouveauté. Lorsque cette nouveauté dépasse un seuil de *frustration*, le changement d'algorithme est activé. Des tests en simulation et avec un robot réel montrent l'efficacité de cette méthode dans le contexte de la robotique.

Renaudo et al. [25], déjà évoqué pour leurs méthodes de fusion, a également testé deux méthodes de sélection par suivi de variables internes. La première consiste à évaluer la récompense moyenne obtenue sur une fenêtre de temps glissante, à activer l'algorithme d'apprentissage sans modèle interne par défaut et à basculer vers l'algorithme avec modèle interne si la récompense moyenne décroît ou stagne. La seconde méthode consiste à mesurer l'entropie des distributions utilisées pour le choix des actions, qui peut être considérée comme une mesure de l'incertitude des algorithmes sur leurs choix. Cependant, aucune de ces deux méthodes n'a été en mesure de faire mieux que l'expérience contrôle.

Enfin, dans le cadre de la modélisation de l'apprentissage d'associations visuo-motrices chez l'humain, Viejo et al. [27] ont repris les idées de [20] et de [24] mettant en balance incertitude et coût d'utilisation de l'algorithme avec modèle interne, mais en utilisant l'entropie de la distribution des actions comme estimateur de cette incertitude. Un algorithme d'apprentissage sans modèle interne est ici combiné avec un algorithme de délibération bayésien (qui modélise une mémoire de travail). C'est la décroissance de l'entropie au fur et à mesure des inférences de ce dernier qui détermine la probabilité de stopper l'inférence. Cette proposition a été confrontée quantitativement aux méthodes de [20] et de [5], et s'est avérée plus efficace pour prédire à la fois les choix et les temps de réponse des sujets humains dans l'expérience considérée.

Les méthodes de sélection d'algorithmes fondées sur le suivi de l'évolution de variables internes semblent en général donner de bons résultats et permettent potentiellement de bénéficier d'économies de ressources de calcul. Il n'y a pas encore de consensus sur la nature des variables à suivre (incertitude, nouveauté, entropie, etc.), mais globalement toutes visent à estimer la performance d'un algorithme à un instant donné et à, alternativement, la comparer à celle de l'algorithme compétiteur ou au coût des calculs qui pourraient être mis en oeuvre pour améliorer cette performance.

## 2.3 Sélection par apprentissage

La sélection par suivi de variables implique que le concepteur de la méthode choisisse par avance quelles seront ces variables, ainsi que comment elles seront exploitées. Il peut être intéressant, dans le cas où l'on recherche une autonomie et une adaptabilité maximale de l'agent, de s'en remettre aussi à l'apprentissage pour déterminer quelles sont les situations ou les indicateurs pertinents pour déclencher un changement d'algorithme actif.

La première méthode proposant d'apprendre de manière autonome à sélectionner entre deux algorithmes de navigation est celle de Foster et al. [11]. Cette méthode issue des neurosciences computationnelles cherchait à expliquer le comportement de rats dans des tâches de navigation. Elle a ceci de particulier qu'elle met en compétition les valeurs *Q(o,a)* issues d'un algorithme d'apprentissage sans modèle interne basé sur la localisation, avec la qualité estimée d'un algorithme de navigation métrique (cherchant à estimer les positions absolues de l'agent et du but). Le niveau de compétition entre algorithmes est donc assez hétérogène, puisque c'est un algorithme entier (de navigation métrique) qui est comparé avec les valeurs individuelles de chacune des actions proposées par un autre algorithme (d'apprentissage sans modèle interne). Cette méthode a permis de montrer que le comportement des rats lorsque le but change de position dans un environnement connu, n'est pas explicable par un algorithme seul d'apprentissage sans modèle interne. Il faut nécessairement lui adjoindre un autre algorithme de navigation.

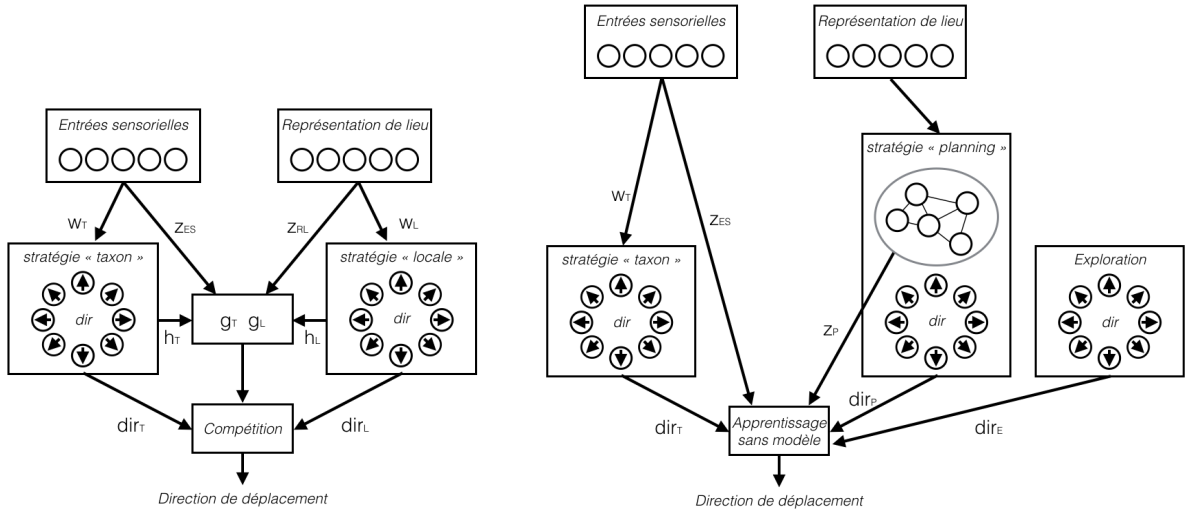


**Figure 7 : Gauche :** méthode de Chavarriaga et al. [4]. **Droite :** méthode de Dollé et al. [8].

Chavarriaga et al. [4] ont également proposé d'expliquer le comportement de rats dans des tâches de navigation en combinant deux algorithmes sans modèle interne, chacun utilisant des données d'entrée différentes pour l'apprentissage (fig. 7, gauche) : l'algorithme d'approche visuelle (dit « *taxon »*) apprend à associer la vision d'indices dans l'environnement (« entrées sensorielles » sur la fig. 7) avec la direction de déplacement (*dir*), alors que l'algorithme d'association lieu-réponse (« *locale »*) s'appuie sur l'estimation de la localisation (« représentation de lieu » sur la fig. 7). Les fonctions de valeur respectives des ces deux algorithmes sont stockées dans les poids $w_T$ et $w_L$. Le signal d'apprentissage dans chaque algorithme est atténué par un facteur supplémentaire *h*, qui dépend de l'erreur de prédiction de récompense $\delta$ de chacun : c'est celui qui fait le moins d'erreur dans une situation particulière qui recevra le signal d'apprentissage le moins atténué, une technique inspirée des mixtures d'experts [17]. Pour ce qui est de la sélection entre algorithmes, elle s'effectue sur la base de valeurs de sélection (*gating values*, $g_T$ et $g_L$) qui s'ajustent progressivement (par modification des poids $z_{ES}$ et $z_{RL}$) pour tendre vers la valeur moyenne de *h* dans chaque situation rencontrée (combinaison de ce qui est vu et de la localisation estimée). La méthode apprend donc progressivement quel est l'algorithme (ou « l'expert » pour parler en terme de mixtures d'experts) le plus efficace dans chaque configuration des données sensorielles. L'une de ses principales limitations est qu'il est entièrement fondé sur les erreurs de prédiction de récompense de chaque algorithme, une variable qui n'est, a priori, calculée que pour les algorithmes d'apprentissage sans modèle interne. La méthode n'est donc pas assez générique pour permettre la combinaison d'algorithmes de nature différente.

Cette méthode ayant néanmoins permis de rendre compte d'un certain nombre d'observations expérimentales, mais n'ayant par ailleurs pas permis de toutes les expliquer, une nouvelle méthode poursuivant les mêmes objectifs a été développée [8]. Dans ce cas-ci, un algorithme d'apprentissage sans modèle interne, réalisant une approche visuelle (similaire au « *taxon »* de Chavarriaga et al. [4]) était combiné avec un algorithme d'apprentissage avec modèle interne, basé sur la localisation (« *planning »* sur la fig. 7, droite, dont le modèle interne est représenté par le graphe entouré d'une ellipse grise). La sélection entre algorithmes était réalisée de manière entièrement autonome par un troisième algorithme d'apprentissage, sans modèle interne, prenant en entrée la concaténation des informations de vision et de localisation et apprenant à choisir (en modifiant les poids $w_{ES}$ et $w_P$) sur cette base quel algorithme activer pour maximiser la récompense (fig. 7, droite). D'un point de

vue calculatoire, ce cadre permet de lever les contraintes sur la nature des algorithmes combinés, puisqu'il utilise une monnaie commune de comparaison indépendantes du type d'algorithme. D'un point de vue modélisation computationnelle, cette méthode a permis de mieux expliquer les données expérimentales considérées chez le rat que Chavarriaga et al. [4]. On notera au passage que, comme dans le système proposé par Girard et al. [12], l'exploration n'était dans cette méthode plus prise en charge par une sélection probabiliste à l'intérieur de chaque algorithme (au lieu d'un *softmax* sur les valeurs Q(o,a), c'est un *argmax* qui était appliqué), mais par un algorithme dédié (« *exploration* » fig. 6, droite), mis en compétition avec les deux autres. Cela permet donc au système de sélection d'algorithme de réguler lui-même la quantité d'exploration en fonction des besoins, réalisant en cela le méta-apprentissage évoqué au paragraphe 1.1.

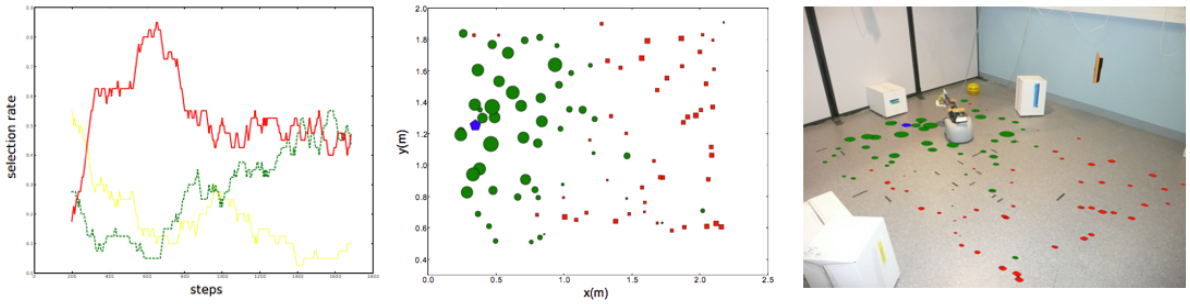


**Figure 8 : Gauche :** Evolution des taux de sélection au cours de l'apprentissage dans la tâche de navigation de Caluwaerts et al. [3]. **Milieu :** Stratégies choisies selon la position dans l'environnement (voir texte pour explications). **Droite :** surimpression du graphe du milieu sur une photo de l'environnement et du robot utilisés.

Enfin, cette même méthode, initialement proposée pour expliquer des données de neurosciences, a été appliquée à la robotique [3] : adaptée pour l'utilisation des capteurs et des actionneurs réels, elle a pu être testée sur une plateforme physique. La figure 8 représente la tâche de navigation utilisée : le robot doit apprendre par essais et erreurs à rejoindre une position où lui sera donné une récompense (représenté par un pentagone bleu au milieu et à droite de la fig. 8). Le graphe de gauche (fig.8) représente l'évolution des taux de sélection, au cours du temps, des trois algorithmes en compétition : on constate que l'exploration (en jaune) domine en début d'expérience, est remplacée par l'apprentissage avec modèle interne (rouge) qui apprend vite à planifier de bonne trajectoires. Enfin, l'approche visuelle est plus lente à apprendre (apprentissage sans modèle) mais finit par être utilisée la moitié du temps. La fig. 8 milieu représente, en fin d'apprentissage, les emplacements de l'environnement où sont utilisés les différents algorithmes (la couleur des symboles indique l'algorithme utilisé préférentiellement et leur taille l'intensité de cette préférence) : à proximité du but, l'approche visuelle est favorisée (cercles verts) ; à distance, là où l'indice visuel est peu visible, l'apprentissage avec modèle interne est préféré car plus efficace (carrés rouges). Cette mise en oeuvre robotique a permis de montrer expérimentalement que cette approche issue des neurosciences, combinant algorithmes de navigation multiples et apprentissage par renforcement pour leur sélection, permettait également d'améliorer les performances d'un robot : l'utilisation des deux algorithmes combinés s'est avérée supérieure à celle de chacun pris séparément. De plus, l'ajout d'un algorithme de détection de changements de contexte a permis d'accélérer l'adaptation du robot à un changement de la position du but. En effet, la méthode proposée en neurosciences par Dollé et al. [8] pour apprendre quel algorithme est le plus efficace dans chaque sous-partie de l'environnement présente l'inconvénient d'être lente à adapter à des changements de

l'environnement. Ainsi, des changements de position du but à atteindre par le robot nécessitent non seulement un ré-apprentissage de chacun des algorithmes en jeu, mais également un ré-apprentissage de la sélection d'algorithme. L'idée nouvelle issue de l'application robotique de cette méthode par Caluwaerts et al. [3] consiste à reconnaître les changements soudains de motif d'activation des valeurs *Q(o,a)* calculées par le module d'apprentissage avec modèle interne comme des signes que l'environnement a changé. Chacune des configurations reconnues de l'environnement (assimilées aux différents motifs d'activation calculés grâce le modèle interne) est considérée comme un contexte différent n'interférant pas avec ce qui a été appris dans les autres contextes. Ainsi, le robot contrôlé par cette méthode a pu rapidement reconnaître un changement de contexte lorsque la position du but change, et ainsi s'adapter plus rapidement. De plus, lorsque la position du but est ramenée à sa position initiale, le robot reconnaît automatiquement le contexte initial et peut ainsi restaurer ce qu'il avait appris dans ce contexte.

## 3. Conclusion

La combinaison de modules multiples pour la résolution d'une tâche en robotique autonome (et en particulier pour la navigation) a été peu explorée jusqu'ici. Pourtant, les complémentarités en matière de vitesse d'adaptation et de coût computationnel des différents types d'algorithmes d'apprentissage par renforcement susceptibles d'être utilisés, d'une part, et les résultats de neurosciences démontrant l'utilisation de systèmes d'apprentissage multiples chez les animaux, d'autre part, plaident en faveur de la conception d'architectures de contrôle intégrant de multiples algorithmes.

C'est cette exploration encore limitée des possibilités de telles combinaisons en robotique qui explique que bon nombre des algorithmes passés en revue ici sont issus des neurosciences computationnelles, et ont pour but initial d'expliquer des données biologiques. Pourtant, ces méthodes, exprimées dans le formalisme de l'apprentissage par renforcement sont parfaitement adaptables et testables en robotique, ce que nous nous sommes efforcés de mettre en évidence.

Comme le souligne un article récent de synthèse des travaux sur l'apprentissage en robotique [23], il n'y a d'une part actuellement pas de solution générale qui puisse permette à un même robot d'apprendre correctement chacun des problèmes étudiés, et il n'y a d'autre part pas de comparaison facile à faire entre toutes ces solutions puisqu'elles n'ont pas été testées sur un ou plusieurs mêmes problèmes. La proposition qui émerge du travail de Caluwaerts et al. [3] consiste à dire que chaque algorithme d'apprentissage a ses avantages et ses inconvénients, et qu'une méthode de coordination par méta-apprentissage peut apprendre automatiquement quel algorithme est avantageux face à chaque problème rencontré. Cela est similaire à ce qu'on observe dans le vivant : les mammifères et en particulier l'humain n'atteignent jamais une performance optimale sur un problème donné. Néanmoins les humains sont capables de s'adapter et d'atteindre une performance satisfaisante face à de nombreux problèmes différents. La coordination d'algorithmes d'apprentissage est donc une idée principalement bio-inspirée qui semble prometteuse pour l'apprentissage en robotique.

La combinaison d'algorithmes multiples peut prendre la forme d'une fusion de leurs sorties (paragraphe 2.1), on est alors dans un cadre qui rappelle les méthodes

d’ensemble de l’apprentissage automatique [7]. Cette approche permet d’améliorer les performances de l’apprentissage, elle a cependant pour inconvénient de ne pas permettre d’économies de ressources de calcul, puisque l’ensemble des systèmes sont calculés.

Les méthodes de sélection (paragraphe 2.2 et 2.3) permettent, elles, théoriquement, de faire de telles économies. On a vu qu’elles ne sont cependant pas toutes conçues pour cela : seuls quelques-unes d’entre elles prennent explicitement en compte les coûts computationnels des algorithmes utilisés [20,24,27], et aucune de celles-ci n’ont été mises en oeuvre dans un cadre robotique. Les méthodes de sélection qui misent sur une conception a priori des variables à observer et des règles à appliquer pour arbitrer les changements de contrôleurs sélectionnés (paragraphe 2.2) cherchent en général à évaluer le degré de confiance que l’on peut avoir dans chaque contrôleur avec des mesures plus ou moins directes de l’incertitude de leurs évaluations. Les méthodes de sélection par apprentissage (paragraphe 2.3) ont l’avantage de ne pas fixer a priori les règles qui seront appliquées pour déterminer les changements de sélection, mais de les découvrir par interactions avec l’environnement, et donc potentiellement de les adapter à chaque cas. Cependant, aucune d’entre elles n’a jusqu’ici intégré explicitement la notion de coût computationnel, ce qui pourrait probablement être réalisé simplement en intégrant en plus des récompenses un coût pénalisant l’utilisation des algorithmes les plus coûteux et favorisant donc les meilleurs compromis précision/ressources.

Enfin, on notera que certaines des méthodes de combinaison d'algorithmes d’apprentissage multiples ne font pas systématiquement usage d’algorithmes issus des deux familles d’apprentissage par renforcement présentées (avec et sans modèle interne de l’environnement) : un même algorithme dupliqué et travaillant avec des données d’origines différentes, permet aussi d’améliorer les performances. Au-delà de cette simple dichotomie avec/sans modèle interne, cela ouvre donc la voie à des algorithmes multiples, comprenant plus de deux algorithmes, intégrant des algorithmes variés, et travaillant sur des données variées.

# Coopération d'algorithmes d’apprentissage par renforcement multiples

*Cooperation of multiple reinforcement learning algorithms*

*par* **Benoît GIRARD**

Directeur de Recherche CNRS

Institut des Systèmes Intelligents et de Robotique, ISIR (UMR7222, CNRS - UPMC)

**Mehdi KHAMASSI**

Chargé de Recherche CNRS

Institut des Systèmes Intelligents et de Robotique, ISIR (UMR7222, CNRS - UPMC)

## Sources bibliographiques

[1] Balleine, B. W., & O'Doherty, J. P. (2010). Human and rodent homologies in action control: corticostriatal determinants of goal-directed and habitual action. Neuropsychopharmacology, 35(1), 48-69.

[2] Bellman, R. E. (1957). Dynamic Programming. Princeton University Press, Princeton, NJ.

[3] Caluwaerts, K., Staffa, M., N’Guyen, S., Grand, C., Dollé, L., Favre-Félix, A., Girard, B. & Khamassi, M. (2012). A biologically inspired meta-control navigation system for the psikharpax rat robot. Bioinspiration & biomimetics, 7(2), 025009.

[4] Chavarriaga, R., Strösslin, T., Sheynikhovich, D., & Gerstner, W. (2005). A computational model of parallel navigation systems in rodents. Neuroinformatics, 3(3), 223-241.

[5] Collins, A. G., & Frank, M. J. (2012). How much of reinforcement learning is working memory, not reinforcement learning? A behavioral, computational, and neurogenetic analysis. European Journal of Neuroscience, 35(7), 1024-1035.

[6] Daw, N. D., Niv, Y., & Dayan, P. (2005). Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. Nature neuroscience, 8(12), 1704-1711.

[7] Dietterich, T. G. (2000). Ensemble methods in machine learning. In Multiple classifier systems (pp. 1-15). Springer Berlin Heidelberg.

[8] Dollé, L., Sheynikhovich, D., Girard, B., Chavarriaga, R., & Guillot, A. (2010). Path planning versus cue responding: a bioinspired model of switching between navigation strategies. Biological cybernetics, 103(4), 299-317.

[9] Dickinson, A. (1985). Actions and habits: the development of behavioural autonomy. Philosophical Transactions of the Royal Society of London B: Biological Sciences, 308(1135), 67-78.

[10] Dolan, R. J., & Dayan, P. (2013). Goals and habits in the brain. Neuron, 80(2), 312-325.

[11] Foster, D. J., Morris, R. G. M., & Dayan, P. (2000). A model of hippocampally dependent navigation, using the temporal difference learning rule. Hippocampus, 10(1), 1-16.

[12] Girard, B., Filliat, D., Meyer, J. A., Berthoz, A., & Guillot, A. (2005). Integration of navigation and action selection functionalities in a computational model of cortico-basal-ganglia–thalamo-cortical loops. Adaptive Behavior, 13(2), 115-130.

[13] Guazzelli, A., Bota, M., Corbacho, F. J., & Arbib, M. A. (1998). Affordances. Motivations, and the World Graph Theory. Adaptive Behavior, 6(3-4), 435-471.

[14] Hanoune, S. (2015). Vers un modèle biologiquement plausible de la sélection de l'action pour un robot mobile (Doctoral dissertation, Université de Cergy-Pontoise).

[15] Hanoune, S., Banquet, J. P., Gaussier, P., & Quoy, M. (2015). Cooperation/ supervision of a habit by a cognitive strategy in a goal-directed navigational paradigm. BMC Neuroscience, 16(Suppl 1), P200.

[16] Hasson, C., Gaussier, P., & Boucenna, S. (2011). Emotions as a dynamical system: the interplay between the meta-control and communication function of emotions. Paladyn, 2(3), 111-125.

[17] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. Neural computation, 3(1), 79-87.

[18] Jauffret, A., Cuperlier, N., Tarroux, P., & Gaussier, P. (2013). From self-assessment to frustration, a small step toward autonomy in robotic navigation. Frontiers in neurorobotics, 7.

[19] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. Journal of artificial intelligence research, 237-285.

[20] Keramati, M., Dezfouli, A., & Piray, P. (2011). Speed/accuracy trade-off between the habitual and the goal-directed processes. PLoS computational biology, 7(5), e1002055.

[21] Khamassi, M., & Humphries, M. (2012). Integrating cortico-limbic-basal ganglia architectures for learning model-based and model-free navigation strategies. Frontiers in behavioral neuroscience.

[22] Killcross, S., & Coutureau, E. (2003). Coordination of actions and habits in the medial prefrontal cortex of rats. Cerebral Cortex, 13(4), 400-408.

[23] Kober, J., Bagnell, J.A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*. 32(11):1238-1274.

[24] Pezzulo, G., Rigoli, F., & Chersi, F. (2013) The Mixed Instrumental Controller: using Value of Information to combine habitual choice and mental simulation. Front. Psychol. 4:92. doi: 10.3389/fpsyg.2013.00092

[25] Renaudo, E., Girard, B., Chatila, R., & Khamassi, M. (2015). Which criteria for autonomously shifting between goal-directed and habitual behaviors in robots? In 5th International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EPIROB).

[26] Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction. MIT press.

[27] Viejo, G., Khamassi, M., Brovelli, A., & Girard, B. (2015). Modeling choice and reaction time during arbitrary visuomotor learning through the coordination of adaptive working memory and reinforcement learning. Frontiers in behavioral neuroscience, 9.

[28] Wiering, M., & Van Hasselt, H. (2008). Ensemble algorithms in reinforcement learning. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 38(4), 930-936.