Approche distribuée de l'apprentissage. Application au contrôle de la trajectoire d'un robot hexapode

Article *in* Journal Européen des Systèmes Automatisés · October 2005

DOI: 10.3166/jesa.39.965-993

CITATIONS
0

READS
103

2 authors:

Zennir Youcef
Université 20 août 1955-Skikda
121 PUBLICATIONS **454** CITATIONS

SEE PROFILE

Pierre Couturier
IMT Mines Alès
47 PUBLICATIONS **125** CITATIONS

SEE PROFILE

Approche distribuée de l'apprentissage : Application au contrôle de la trajectoire d'un robot hexapode

Youcef Zennir, Pierre Couturier

Centre de recherche LGI2P, EMA site EERIE
Parc Scientifique Georges Besse 300035 Nimes Cedex 1

Pierre.Couturier@ema.fr *Youcef.Zennir@ema.fr*

RÉSUMÉ. *Une approche distribuée de l'apprentissage par renforcement de type Q-learning est proposée dans laquelle des groupes d'agents contribuant à une même tâche mènent leur apprentissage en tenant compte ou non de l'existence des autres agents. L'approche est appliquée à la commande d'un robot hexapode pour qu'il apprenne à marcher et à changer de trajectoire tout en contrôlant sa posture. Les résultats de simulation valident l'approche proposée. D'autres travaux de recherche relatifs au domaine sont discutés.*

ABSTRACT. *A distributed approach of the Q-learning technique is investigated so as groups of agents participating to the same global task perform their learning process taking into account or not the other agents. The approach is applied to a hexapod robot that learns to walk and to change of trajectory while controlling its posture. The efficiency of the approach is validated through simulation results. Some related research works are discussed.*

MOTS CLÉS: *Apprentissage par renforcement, Q-learning, Systèmes distribués, Robot hexapode, Contrôle de la posture*

KEYWORDS: *Distributed reinforcement learning, Q-learning, Hexapod robot, Posture control*

1. Introduction

La problématique de la coopération dans les systèmes multi-agents constitue un domaine de recherche très actif. Dans de tels systèmes, les agents doivent discerner leur contribution à une tâche commune afin de corriger efficacement leur propre comportement.

La coopération dans les systèmes multi-agents, repose sur la communication, ou sur l'établissement de conventions, ou sur l'apprentissage. Les agents réactifs (nommés ci-dessous 'acteurs') considérés dans cet article agissent en réponse à des stimuli reçus de l'environnement. Ces acteurs disposent d'information d'état sur le système, mais, contrairement aux agents cognitifs, ils ne peuvent ni communiquer ni établir de convention entre eux. Dans ce cadre, l'apprentissage est donc préconisé, l'apprentissage par renforcement étant tout particulièrement indiqué en environnement incertain.

L'application traitée est l'apprentissage de la marche d'un robot hexapode, considéré comme un système distribué dans lequel les acteurs sont les contrôleurs de mouvement de chaque patte.

Plusieurs biologistes ont établi que le contrôle de la marche de l'insecte 'phasme' n'est pas centralisé, mais se décompose en plusieurs sous systèmes indépendants et connectés entre eux (Cruse 1976, Dean 1991). Aussi notre approche consiste à évaluer des modèles de décision non centralisée pour commander les mouvements de locomotion du robot hexapode.

L'objectif de ce travail est d'étudier une méthode d'apprentissage de la marche à faible allure avec maîtrise de la trajectoire et de la posture. Un déplacement à faible allure permet de négliger les effets dynamiques des masses en mouvement. Les fonctions à assurer sont :

- **le maintien de l'équilibre :** en négligeant les effets dynamiques, le maintien de l'équilibre se ramène à la satisfaction de la stabilité statique. Ainsi, sur terrain plat, deux pattes consécutives ne peuvent pas être en l'air simultanément,
- **le contrôle de la posture :** il s'agit de maintenir la position relative du corps par rapport aux pieds en contact avec le sol proche d'une position de référence offrant le meilleur compromis entre mobilité et stabilité,
- **la génération de mouvements de marche :** le cycle locomoteur d'une patte définit l'ensemble des événements articulaires qui se produisent entre deux appuis successifs au sol. Pour générer un mouvement de marche, il faut coordonner les cycles locomoteurs des pattes,
- **le changement de trajectoires :** le robot doit être capable de changer de position et d'orientation soit pour contourner un obstacle, soit pour se diriger vers un point d'intérêt de son environnement.

Après avoir situé, en section 2, notre approche parmi les travaux récents sur l'apprentissage en environnement distribué et sur la robotique hexapode, nous décrivons en section 3 une implémentation distribuée du Q-learning. Selon une stratégie individuelle, les acteurs ignorent l'existence des autres acteurs. Selon une stratégie collective, chaque acteur tient compte a posteriori des actions des autres acteurs impliqués dans la tâche commune. Une présentation basée sur la notion de groupe d'acteurs capables de s'observer, est proposée à la section 4. Un exemple de stratégie individuelle est illustré section 5 pour l'apprentissage de la marche : Chaque patte est considérée comme un acteur autonome, devant reproduire un cycle locomoteur compatible avec ceux des autres pattes. La réalisation des cycles locomoteurs individuels produit une marche périodique. Un exemple de stratégie collective est illustré à la section 6 par des simulations menées pour apprendre au robot marcheur à changer de trajectoire tout en contrôlant sa posture. A cette fin, les acteurs sont répartis en deux groupes indépendants.

2. Apprentissage d'agents réactifs et robotique hexapode

2.1. Apprentissage d'agents réactifs et coopératifs

On peut distinguer trois approches pour concevoir un système d'agents réactifs capables d'apprendre collectivement à réaliser une tâche commune.

Une première approche, contourne le problème de coopération et fixe des objectifs locaux et individuels compatibles avec l'objectif global. L'acteur apprend à résoudre la tâche sans tenir compte des autres. Les travaux de K. Tumer et D. Wolpert tendent à établir une théorie mathématique permettant de construire de tels objectifs locaux compatibles avec l'objectif global (Tumer *et al*., 2000).

Une deuxième approche consiste à observer le comportement des autres acteurs pour déterminer une politique locale efficace. Ainsi, supposant connus les gains espérés, C. Clauss et C. Boutillier proposent que chaque acteur estime les probabilités de sélection des actions par les autres acteurs. Un acteur peut alors anticiper les actions des autres et agir en concordance (Clauss *et al*., 1998). Dans le cas fréquent où les gains espérés par les acteurs ne sont pas connus a priori, J. Hu et M. Wellman proposent l'algorithme Nash-Q-learning (Hu *et al.,* 1998) dans lequel chaque acteur estime l'espérance de ses gains et celle des gains des autres acteurs tout en améliorant sa propre stratégie. Les acteurs sont considérés comme des joueurs participant à un jeu stochastique et choisissent des actions correspondant à un équilibre de Nash (état d'équilibre dont aucun des acteurs n'a intérêt à dévier de façon unilatérale). Malgré l'établissement de certaines preuves de convergence, la méthode est gourmande en mémoire et en temps de calcul et s'avère alors peu exploitable lorsque le nombre d'acteurs, de situations ou d'actions croît. Le choix coordonné entre plusieurs équilibres de Nash pose aussi problème. Dans les cas où

la convergence de Nash-Q learning a été prouvée, d'autres méthodes d'apprentissage ne supposant pas la construction de modèle du comportement des autres acteurs tels que le minimax Q-learning et le Team Q-learning proposés par M.L. Littman sont aussi applicables (Littman, 2001).

Une troisième approche consiste à construire un modèle de transition d'état et à l'exploiter pour que chaque acteur, à partir d'un ensemble d'observations locales, puisse ordonner ses actions par ordre d'utilité et adopter celle qui limite le risque d'échec. Cette approche est basée, d'une part sur 'l'apprentissage par le cas' ('lazy learning') pour l'établissement d'un modèle de transition suffisamment riche pour permettre un apprentissage et d'autre part sur un algorithme 'pessimiste' pour la prise de décision (Touzet, 2004). L'idée de cet algorithme est que la contribution propre de l'acteur à la tâche apparaît au moins dans une des situations voisines d'une situation donnée et qu'en sélectionnant la « moins défavorable » des actions évaluées dans ces situations voisines, l'acteur limite le risque de recevoir une pénalité. Cette approche a été validée dans le contexte de robots coopérant à la surveillance de cibles mouvantes.

Nos travaux s'apparentent à la seconde approche dans laquelle plusieurs groupes d'acteurs indépendants apprennent à résoudre une tâche commune. Au sein de chaque groupe, les acteurs disposent d'information sur les agissements des autres mais n'élaborent pas de modèle comportemental des autres acteurs.

2.2. La robotique hexapode

Les robots marcheurs à six pattes (robots hexapodes) font l'objet de nombreux travaux de recherche car ils présentent un bon compromis entre complexité de conception, stabilité et mobilité (Brooks, 1989), (Kirchner, 1998), (Kingsley *et al*., 2003). Ils peuvent être utiles pour accéder à des zones hostiles pour l'homme comme les sites nucléaires ou l'espace. Ces robots possèdent un nombre élevé de degrés de liberté ($\geq$ 2 par patte) et leur commande nécessite de résoudre simultanément des problèmes de coordination de mouvements, de contrôle de l'équilibre et de contrôle de la posture. Les travaux de recherche portent sur la structure mécanique, l'élaboration de modèles géométriques, cinématiques ou dynamiques ainsi que sur les méthodes de commande du robot. Dans un environnement inconnu ou incertain, la planification des mouvements n'est pas efficace. Il est alors intéressant de recourir à un apprentissage tel que l'apprentissage par renforcement qui se base sur un processus essai / erreur pour acquérir le comportement souhaité. La résolution de tels problèmes d'apprentissage est menée à partir de modèles simplifiés des robots hexapodes décrivant la séquence des mouvements et la marche obtenue.

C. Touzet et A. Johannet (Touzet, 1992), (Johannet, 94) ont proposé une architecture de commande distribuée à base de six réseaux de neurones

stochastiques pilotant les mouvements de poussée ou de transfert de chaque patte. L'apprentissage est mené selon l'agorithme de 'pénalité / récompense' de G. Barto (Barto, 1985). Le robot dont un prototype physique a été réalisé (poids 3 kg, dimension (en mm) : 300x150x150), apprend à marcher et à éviter des obstacles.

Constatant qu'un robot, pour évoluer dans un monde réel, a besoin d'un grand nombre de capteurs mais que tous ne fournissent pas des informations pertinentes pour la tâche à effectuer, J.M. Porta (Porta, 2000) introduit un nouvel algorithme d'apprentissage, le 'ρ learning' dont l'objectif est de diminuer le temps nécessaire à l'apprentissage de la marche d'un robot hexapode en découvrant les sous-ensembles de capteurs qui donnent les informations utiles pour la tâche. La méthode a été validée en simulation.

Les travaux de M. M. Svinin (Svinin, 2001) portent sur la commande d'un robot marcheur selon une approche à base de règles et d'apprentissage par renforcement. Ce robot n'a pas de connaissance a priori sur l'environnement et cherche à atteindre un point particulier de l'espace. Une règle associe un vecteur d'entrée (composé de valeurs particulières des capteurs du robot) à une action. L'apprentissage consiste à sélectionner les règles les plus pertinentes pour atteindre l'objectif. Il est montré en simulation que le robot apprend à marcher en direction de l'objectif sans tomber mais qu'il ne suit pas forcément le chemin le plus court.

Les travaux de C. Touzet et A. Johannet ont permis de montrer qu'un robot hexapode avec une architecture de commande distribuée pouvait appendre à marcher. Cependant l'attribution des pénalités et des récompenses n'est pas locale, les pénalités sont appliquées à tous les contrôleurs de mouvement y compris à ceux qui ont pris une décision correcte. Dans la suite sont présentés des résultats montrant l'intérêt de récompenser ou de pénaliser individuellement les acteurs ; à la section 6, le modèle de simulation est enrichi d'un modèle géométrique de contrôle de la posture pour apprendre au robot à maîtriser sa trajectoire.

3. Approches centralisée et distribuée du Q-learning

3.1. Apprentissage par renforcement et Q-learning

D'après S. Sutton et G. Barto (Sutton et Barto, 1998), l'apprentissage par renforcement définit un type d'interaction entre l'acteur et l'environnement. Depuis une situation réelle « **s** » dans l'environnement, l'acteur choisit et exécute une action «**a**» qui provoque une transition vers un état «**s'**». Il reçoit en retour d'un critique un signal de renforcement «**r**» négatif et de type pénalité si l'action conduit à un échec, positif et de type récompense si l'action est bénéfique, nul s'il n'est pas possible d'attribuer une pénalité ou une récompense.

L'objectif de l'acteur est de maximiser à tout instant t la somme pondérée R_t de ses gains futurs :

$R_t = r_t + \gamma.r_{t+1} + \gamma^2.r_{t+2} + ...$ avec $0 \leq \gamma \leq 1$

Les algorithmes d'apprentissage par renforcement sont basés sur le calcul de l'une des deux fonctions d'utilité V ou Q suivantes :

$V^{\pi}(s)$, (respectivement $Q^{\pi}(s,a)$) est l'espérance mathématique de la somme des récompenses futures dans l'état s (respectivement dans l'état s et en exécutant l'action a) en appliquant une stratégie π de sélection d'actions :

$$V^{\pi}(s) = E_{\pi}\{R_t / s_t = s\},\ Q^{\pi}(s,a) = E_{\pi}\{R_t / s_t = s, a_t = a\} \qquad [1]$$

L'apprentissage par renforcement nécessite de trouver un compromis entre les phases d'exploitation pendant lesquelles l'acteur choisit parmi les actions déjà expérimentées celles qui maximisent ses gains futurs et les phases d'exploration pendant lesquelles l'acteur tente un autre choix d'action qui pourrait lui assurer des gains supérieurs.

Nous avons choisi l'apprentissage de type Q-learning proposé par Watkins (Watkins, 1989) et basé sur l'amélioration itérative d'une stratégie d'action π conduisant à la stratégie d'action optimale π^*.

Un modèle du Q-learning représenté en Figure 1 met en évidence les fonctions suivantes (Sehad, 1996) :

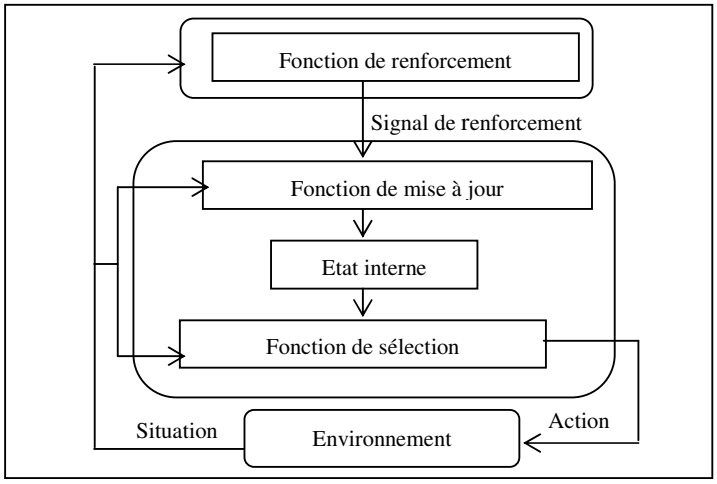

Figure 1. *Processus d'apprentissage par renforcement*

- une fonction de sélection : à partir de la situation actuelle, une action est choisie et exécutée en se basant sur la connaissance disponible au sein de la mémoire interne (cette connaissance est stockée sous forme de valeurs d'utilité),

- une fonction de renforcement (ou fonction critique) : après l'exécution de l'action dans le monde réel, une nouvelle situation est atteinte et cette fonction génère alors la valeur de renforcement,

- une fonction de mise à jour : utilise la valeur de renforcement pour ajuster la fonction d'utilité. L'actualisation à chaque itération de la valeur d'utilité Q(s,a) s'écrit :

$$Q(s,a) \leftarrow Q(s,a) + \alpha.\left[r + \gamma.\max_{a'} Q(s',a') - Q(s,a)\right] \quad \textbf{[2]}$$

avec $\alpha > 0$, $0 \leq \gamma \leq 1$, « s' » l'état suivant, et « a' » l'action suivante.

3.2. Approche centralisée du Q-learning dans le cas de plusieurs acteurs

Considérons N acteurs devant apprendre une stratégie conjointe (la stratégie conjointe désigne l'ensemble des stratégies suivies par tous les acteurs) pour contribuer à une tâche commune. Avec une approche centralisée de l'apprentissage par renforcement, les informations d'état convergent vers un seul centre de décision qui met à jour les valeurs d'utilité et décide des actions de chacun des acteurs.

En désignant par |A| le nombre d'actions exécutables par chaque acteur (nombre supposé identique pour tous les acteurs sans perte de généralité), et |S| le nombre d'états atteignables par le système, il faut maintenir N tables de valeur Q(s, a) de taille $|S|*|A|^N$.

Si on suppose qu'un même signal de renforcement est attribué à tous les acteurs quelle que soit la contribution de chacun au succès ou à l'échec de la tâche commune, alors le problème à résoudre est celui d'un jeu d'équipe stochastique dans lequel il n'y a plus qu'une table Q(s,a) à évaluer.

Les avantages attendus de l'approche centralisée sont :

- une vision globale de l'ensemble du système, permettant d'examiner l'ensemble des situations et des actions possibles,

- les problèmes éventuels de coordination entre acteurs sont résolus à un niveau unique de décision.

Cette approche présente les inconvénients suivants :

- le système global est sensible à la défaillance du centre de décision unique,

- le nombre de couples (s, a) croît exponentiellement avec le nombre d'acteurs,

- pour les jeux d'équipes, les actions bénéfiques au niveau du système global peuvent être pénalisantes pour un acteur, car il n'est pas tenu compte des contraintes locales.

3.3. Approche distribuée du Q-learning dans le cas de plusieurs acteurs

Avec une approche distribuée de l'apprentissage, illustrée par la Figure 2, les acteurs ne reçoivent pas nécessairement la même information d'état ni le même signal de renforcement, et mènent leur propre apprentissage. En supposant que les N acteurs partagent la même information d'état, le nombre de valeurs Q(s, a) à actualiser est N*|S|*|A|.

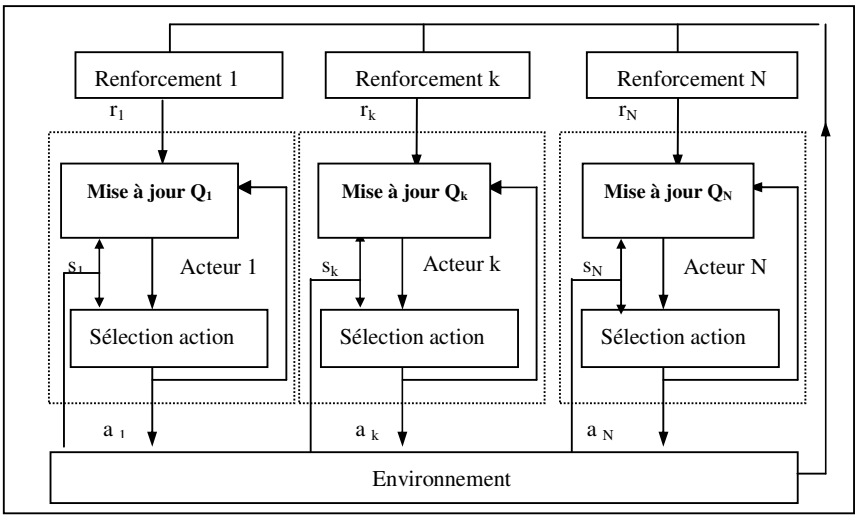

Figure 2. *Architecture distribuée de l'apprentissage par renforcement*

Les avantages attendus de cette approche distribuée sont :

- une plus grande flexibilité (facilité d'adaptation aux modifications imprévues de l'environnement),
- une plus grande fiabilité (l'erreur individuelle est tolérée),
- une plus grande robustesse (la capacité de résolution résulte du collectif et non pas d'un individu),
- chaque acteur peut être soumis à des contraintes locales,
- le nombre de couples (s, a) croît proportionnellement au nombre d'acteurs.

Mais cette approche présente les difficultés suivantes :

- dans le cadre d'une tâche collective les conséquences de l'action d'un acteur dépendent des actions des autres acteurs,

- les objectifs locaux poursuivis par chaque acteur doivent être compatibles avec l'objectif global,

- des mécanismes de coopération parmi lesquels, la synchronisation (enchaînement des actions dans le temps), la collaboration (le partage des tâches), la coordination (la résolution de conflits, l'augmentation des performances du collectif) peuvent être nécessaires.

Une formulation basée sur la notion de groupe d'acteurs et réalisant un compromis entre l'approche centralisée et l'approche distribuée est proposée dans le paragraphe suivant.

4. Q-learning et groupes d'acteurs

4.1. Stratégies individuelle et collective

Dans le cas d'une approche distribuée de l'apprentissage par renforcement, les stratégies suivies par les acteurs impliqués dans une même tâche peuvent être individuelles ou collectives. Selon une stratégie individuelle chaque acteur mène son apprentissage en ignorant les autres acteurs. Cela revient à appliquer l'algorithme d'apprentissage comme si chaque acteur était seul. L'environnement de l'acteur est alors non stationnaire car au cours de l'apprentissage, une action «a » exécutée depuis le même état «s » ne conduit pas toujours au même état «s' », l'état atteint dépendant des actions des autres acteurs. Selon une stratégie collective l'acteur prend en compte l'existence des autres mais peut adopter soit un comportement égoïste (l'acteur cherche à maximiser ses propres gains) soit un comportement altruiste (l'acteur agit pour augmenter aussi les gains d'autres acteurs). Ainsi un comportement conduisant à un équilibre de Nash est tel qu'aucun des acteurs ne peut espérer un gain supérieur compte tenu du choix des actions des autres. Tout équilibre de Nash n'est pas forcément optimal au sens de Pareto, c'est-à-dire qu'il peut exister un autre choix d'action pour un des agents conduisant à des gains non inférieurs pour tous et supérieurs pour au moins l'un d'entre eux. Un équilibre de coordination est un équilibre optimal au sens de Pareto dans lequel chaque agent reçoit le gain maximum.

L'exemple de la Figure 3 illustre divers comportements dans le cas de deux acteurs : chaque matrice représente les gains attendus en fonction des actions choisies par chacun d'eux depuis une situation s. Selon un comportement égoïste, le meilleur choix de l'acteur 1 est a^1_1 et son gain attendu est 3 tandis que le meilleur choix de l'acteur 2 est a^2_3 et son gain attendu est 9. Cependant en exécutant ces actions, le gain obtenu par l'acteur 1 est en fait - 3, la somme des gains des deux acteurs est égale à 6. Selon un comportement altruiste, si l'acteur 1 choisit a^1_2 et l'acteur 2 choisit a^2_2 alors il y a équilibre de Nash (aucun des acteurs n'a intérêt à modifier unilatéralement son choix), la somme des gains est égale à 2. Avec le choix (a^1_3, a^2_1), tous deux obtiennent des gains positifs et la somme de ces gains est 5,

l'équilibre est optimal au sens de Pareto. Notons que le choix (a^1_1, a^2_3) qui maximise la somme des gains n'est pas satisfaisant pour l'acteur 1 qui connaît une perte de 3.

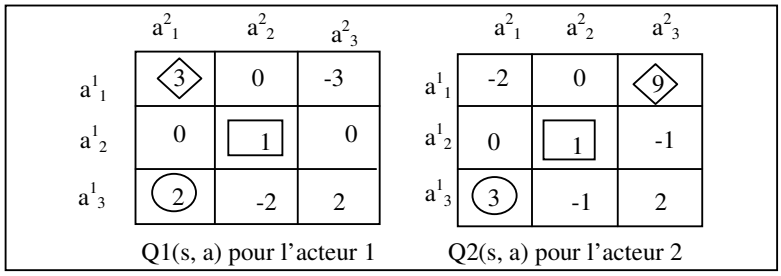

Figure 3. *Matrices de valeurs de Q pour deux acteurs dans l'état s*

Cet exemple montre que les acteurs peuvent avoir intérêt à coopérer plutôt qu'à s'ignorer. Cependant la stratégie de coopération à adopter est différente selon qu'il s'agit de maximiser la somme des gains, d'assurer une optimalité au sens de Pareto ou bien de satisfaire simplement une condition d'équilibre de Nash. Le raisonnement ci-dessus suppose que chaque acteur possède une connaissance globale de la situation, c'est-à-dire qu'il a accès à l'ensemble des tables des gains attendus par les acteurs. Or, dans le cas de l'apprentissage d'une tâche collective dans un système distribué, ces conditions ne sont pas forcément remplies dans la mesure où :

- la table de valeur d'utilité n'est qu'une estimation de l'espérance du cumul des gains attendus par un acteur. Elle peut comporter des erreurs en début d'apprentissage et évolue au cours du temps vers une estimation correcte (s'il y a convergence de l'algorithme d'apprentissage),

- un acteur ne connaît pas forcément les tables de valeurs d'utilité des autres acteurs. Dans ce cas il doit non seulement calculer sa propre table mais aussi estimer celles des autres. A cette fin il doit connaître les situations perçues par chacun des autres acteurs, les actions choisies par eux et les récompenses ou pénalités obtenues,

- un acteur ne connaît pas forcément les stratégies suivies par les autres acteurs. Il doit alors les observer en permanence pour adapter sa propre stratégie,

- pour résoudre des tâches complexes, il peut être intéressant d'introduire des notions d'apprentissage progressif, le nombre d'agents et la difficulté des tâches à résoudre croissant progressivement (Buffet, 2003).

Il apparaît donc que la mise en oeuvre d'un apprentissage par renforcement dans le cas de systèmes multiacteurs posent des problèmes théoriques et pratiques importants. Dans le paragraphe suivant différents groupes d'acteurs apprennent à résoudre une même tâche. Les membres d'un même groupe prennent en compte les choix d'actions a posteriori des autres membres sans toutefois mettre en oeuvre ni

des procédures d'estimation des tables des valeurs d'utilité des autres membres ni des procédures d'estimation de la probabilité du choix de leurs actions.

4.2. Groupes d'acteurs

On suppose qu'au moment d'agir les acteurs ne connaissent pas a priori les actions sélectionnées par les autres mais que dans un groupe donné d'acteurs, chacun d'entre eux connaît a posteriori les actions exécutées par les autres membres du groupe. Dans le cadre d'une stratégie collective (avec comportement égoïste) nous décrivons un algorithme qui permet aux acteurs des groupes de tenir compte des actions sélectionnées par les autres membres d'un même groupe. Avec quelques restrictions énoncées plus bas, cet algorithme appliqué à un seul groupe est l'algorithme 'Team Q-learning' de M.L. Littman (Littman, 2001). Le principe est présenté sur l'exemple simple suivant :

Considérons trois acteurs i, j, k disposant chacun de deux actions possibles notées 0 et 1. Pour chaque état atteint, les acteurs maintiennent 4 tables de valeurs Q correspondant aux quatre possibilités d'action des deux autres acteurs (Figure 4). Chaque acteur i choisit l'action conduisant à une espérance de gain maximum, c'est-à-dire celle qui, pour un état donné, correspond à la ligne comportant la valeur de gain la plus grande (le gain 5 dans la Figure 4) que les autres acteurs choisissent ou non les actions correspondant à la colonne comportant cette valeur. Cependant, à la réception du signal de renforcement, l'acteur met à jour la valeur correspondant aussi au choix d'action des autres acteurs. Ainsi dans l'exemple de la Figure 4, si l'acteur j exécute l'action 0 et l'acteur k l'action 1, c'est la valeur Q = - 2 intersection entre la ligne $a^i = 1$ et la colonne $a^j = 0$, $a^k = 1$ qui est modifiée.

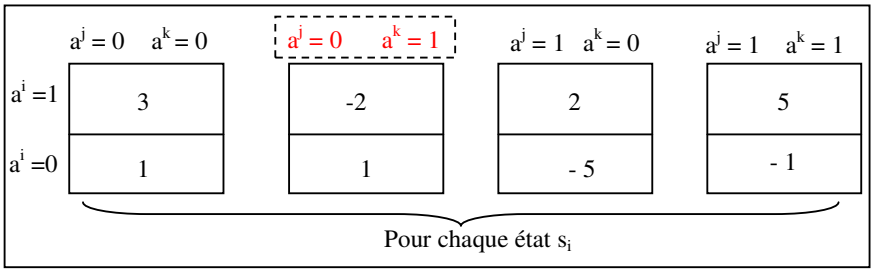

Figure 4. *Valeurs Q pour l'acteur i dans l'état s_i avec 3 acteurs i , j, k et 2 actions*

La valeur de Q qui est actualisée à chaque itération est celle qui correspond aux actions réellement exécutées. Selon cette approche, pour chaque membre i d'un groupe déterminé de K acteurs, l'algorithme Q-learning est décrit Figure 5.

Initialiser $Q_i(s_i, a_1, ...,a_{i-1}, a_i, a_{i+1}, ... a_K)$ *à 0 pour tout i*

Pour tout épisode

Pour tout étape de l'épisode

Pour tout acteur i du même groupe

Choisir a_i *d'après* Q_i *(choix* ε *- glouton)*

Observer $a_1,..,a_{i-1}, a_{i+1}, ... a_K$, s_i' *et* r_i

Réactualiser :

$$Q_i(s_i, a_1,..,a_{i-1}, a_i, a_{i+1}, ... a_K) \leftarrow Q_i(s_i, a_1,..,a_{i-1}, a_i, a_{i+1}, ... a_K) +$$

$$\alpha_i.[r_i+\gamma_i.max_{a'1,...,a'K} Q_i(s_i', a'_1,..., a'_{i-1},,a'_i, a'_{i+1},... a'_K) - Q_i(s_i, a_1,..,a_{i-1}, ,a_i, a_{i+1}, ... a_K)]$$

$$s_i \leftarrow s_i'$$

Figure 5. *Algorithme d'apprentissage multiacteur*

Un choix d'action ε-glouton signifie que l'action correspondant à un gain espéré maximum est choisie avec la probabilité (1-ε), un choix aléatoire d'action étant effectué avec une probabilité ε.

Si chaque groupe ne contient qu'un acteur (K=1), l'algorithme apprend selon une stratégie individuelle (Q-learning de base). Dans un groupe, les acteurs résolvent un jeu d'équipe comme défini au paragraphe 3.2 à condition que :

- les états soient les mêmes pour tous les membres du groupe,
- le signal de renforcement soit le même pour tous les acteurs,
- en cas de choix parmi des actions de même utilité, une règle soit convenue pour que les acteurs exécutent la même action conjointe,
- les K tables Q soient initialisées de façon identique,
- les paramètres α et γ de la fonction de mise à jour soient identiques pour tous les acteurs.

Si ces conditions sont respectées, l'algorithme d'apprentissage de la Figure 5, dans le cas d'un seul groupe d'acteurs, est le Team Q-learning de M.L. Littman. Il suffit alors pour mener l'apprentissage de maintenir qu'une seule table de valeur Q. Un acteur maximise les gains de l'équipe en maximisant ses propres gains. Les conditions de convergence de l'algorithme d'apprentissage dans le jeu d'équipe ont été établies (Littman, 2001). Si une des conditions énoncées n'est pas respectée, au cours du temps, les valeurs de Q_i pour des situations identiques peuvent différer selon l'acteur i. L'algorithme s'avère efficace dans le cas où les renforcements ne sont pas identiques pour les membres du groupe mais où il existe un équilibre de coordination pour chaque état (Littman, 2001). La sélection par les acteurs du même

équilibre parmi de multiples équilibres de coordinations constitue un sujet de recherche (Wan, 2003). Dans le cas général, il n'y a pas, à notre connaissance, de preuve de convergence le l'algorithme présenté Figure 5. L'augmentation du nombre de groupes réduit le nombre de valeurs de Q à actualiser. Par exemple, si |S| est le nombre d'états, |A| le nombre d'actions que chaque acteur peut exécuter, M le nombre de groupes et K le nombre de membres dans chaque groupe, le nombre de valeurs de Q est $M*|S|*|A|^K$ (avec N=M*K). Dans le cas où tous les acteurs suivent une stratégie individuelle, le nombre de valeurs de Q est seulement N*|S|*|A|. Le coût de cette réduction du nombre de valeurs de Q est la perte de coordination entre les groupes. Notons que la diminution du nombre de valeurs Q ne signifie pas que l'apprentissage soit systématiquement plus rapide car, d'une part les groupes ne sont pas coordonnés, et d'autre part une des conditions de convergence du Q-learning est que les couples (état, action) soient visités un très grand nombre de fois (le nombre de ces combinaisons est indépendant du nombre de groupes). Dans la section 5, est mis en évidence un cas où l'approche distribuée avec stratégie individuelle est plus performante que l'approche centralisée. Dans la section 6, est présenté un cas où la stratégie individuelle s'avère moins efficace qu'une stratégie collective : deux équipes indépendantes apprennent à résoudre convenablement une tâche commune.

5. Apprentissage de marches périodiques

5.1. Modèle de simulation

Nous donnons un exemple d'application de l'approche distribuée avec stratégie individuelle pour l'apprentissage de marches périodiques d'un robot hexapode. Le robot est décrit par le modèle suivant : chaque patte peut effectuer deux mouvements, une proaction (balancement de l'arrière vers l'avant de la patte au - dessus du sol) et une rétraction (poussée de la patte sur le sol). Dans ce modèle élémentaire, ne sont décrites ni l'orientation du robot, ni l'amplitude du déplacement obtenu. Les effets dynamiques sont négligés. L'accent est mis uniquement sur le séquencement des mouvements qui évitent les chutes et conduisent à des marches périodiques. La durée d'une proaction considérée comme constante est choisie comme unité de temps. La vitesse de la marche est alors directement liée au temps de rétraction, comme il a été observé chez les insectes (Graham, 1985). L'état du robot dans l'environnement est décrit par six variables binaires traduisant la position de chaque patte sur ou au-dessus du sol. Il y a donc 2^6 états possibles. Les mouvements de proaction et de rétraction sont les deux actions possibles de chaque patte. Si deux pattes voisines sont levées simultanément, le robot est dans une configuration instable et il est admis qu'il chute. Après une chute, le robot est re-initialisé dans une configuration stable quelconque.

Chaque acteur (contrôleur de mouvement de patte) mène son apprentissage de type Q-learning (selon une stratégie individuelle) sur la base de l'actualisation d'une table de 64x2 valeurs Q(s, a).

La fonction de renforcement pour chaque patte est la suivante :

Pénalité : r = - 1 si	**Récompense : r = +1 si**
Chute lors d'une proaction	Pas de chute lors de la proaction
L'ordre de proaction est répété deux fois de suite.	Le mouvement de rétraction a dépassé la durée minimale Dmin
Le mouvement de rétraction a une durée trop longue (>Dmax) ou trop courte (<Dmin)	
Toutes les pattes sont au sol	
Les mouvements de proaction se propagent de l'avant vers l'arrière	

Tableau 1. *Règles appliquées par le critique (fonction de renforcement)*

Une des règles de la fonction de renforcement tient compte des mouvements de plusieurs pattes et adresse une pénalité lorsqu'un mouvement de proaction s'effectue alors que la patte collatérale antérieure vient d'effectuer une proaction (les mouvements de proaction se propagent chez les insectes de l'arrière vers l'avant; le cas de la marche tripode faisant exception (Randall, 1999)).

Pour tenir compte de la symétrie du robot et du rôle similaire que jouent les différents pattes lors d'une marche rectiligne sur sol plan, les pattes controlatérales (en vis à vis par rapport à l'axe de symétrie longitudinal) réagissent de façon analogue (c'est-à-dire qu'elles partagent les estimations des valeurs d'utilité Q associées aux couples (s, a) similaires).

Ainsi, avec ces règles, toutes les pattes ne reçoivent pas simultanément le même signal de renforcement.

5.2. Résultats obtenus

Chaque acteur mène un apprentissage de type Q-learning selon une stratégie individuelle. Les paramètres de l'apprentissage sont les suivants :

Paramètres	Description	Valeur
γ	Facteur de pondération	0.95
α	Pas d'apprentissage	0.01
Dmax	Durée maximale de la rétraction	3
Dmin	Durée minimale de la rétraction	Dmax

Tableau 2. *Paramètres de l'apprentissage de la marche*

Chaque essai consiste en un seul épisode qui se termine soit lorsqu'une marche périodique stable (deux pattes consécutives ne sont jamais en l'air simultanément) a été trouvée soit lorsque le nombre d'itérations de l'épisode atteint une valeur maximale. A chaque itération, les contrôleurs prennent une décision de proaction ou de rétraction. Sur 50 épisodes d'au maximum 2000 itérations, des marches périodiques à quatre temps ont été obtenues 33 fois selon trois chronogrammes différents dont un est représenté en Figure 6. Le nombre moyen d'itérations pour obtenir une marche est de 297 itérations et le nombre moyen de chutes est de 40 chutes par épisode se terminant par une marche.

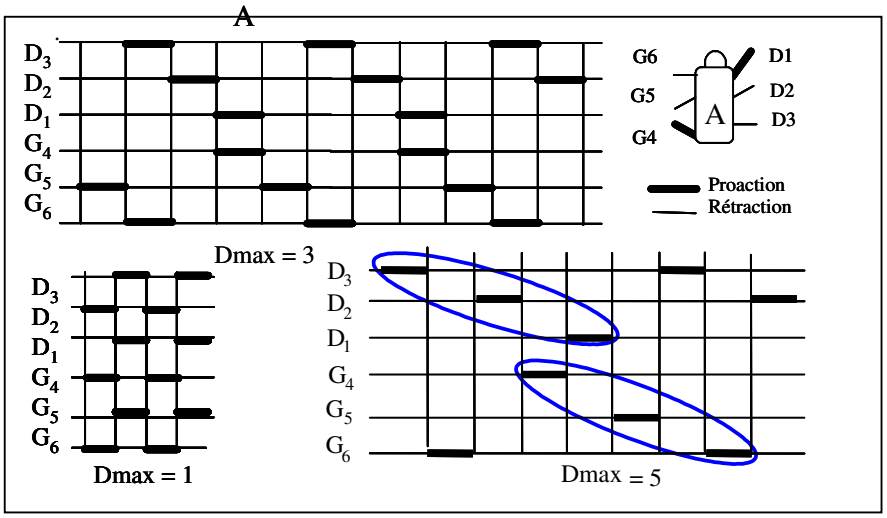

Figure 6. *Chronogrammes de marches périodiques obtenues en fonction de Dmax*

Nous avons pu constaté qu'en faisant varier Dmax entre 1 et 5, il est possible d'obtenir des marches de périodes différentes (Figure 6) : la marche tripode des insectes est obtenue pour une valeur Dmax = 1, une marche à six temps (correspondant au temps maximal de rétraction possible de 5 unités de temps pour chaque patte) est obtenue pour Dmax = 5. Les marches apparaissent comme des cycles stables dans l'espace des configurations. En fin d'apprentissage, lorsque le robot est initialisé selon une configuration stable quelconque, les premiers pas permettent d'atteindre un cycle de marche. Si le paramètre Dmin est choisi inférieur au paramètre Dmax, on trouve des marches de période comprise entre (Dmin+1) et (Dmax+1). En figure 7 est représenté un exemple d'évolution des fonctions Q en début d'apprentissage pour un même état (cas Dmax=3). A partir de la 60 ^{ième} itération, le choix de l'action à effectuer depuis l'état considéré est définitif. Seule la patte G5 quitte le sol. La fonction Q pour l'action choisie converge vers $1/(1-\gamma)$ (dans ce cas la valeur est atteinte à 2% près en 1700 itérations) qui est la valeur de convergence de R_t, somme pondérée des renforcements, lorsque l'acteur ne reçoit que des récompenses de valeur 1. La décision définitive est prise alors que les fonctions Q n'ont pas encore convergées.

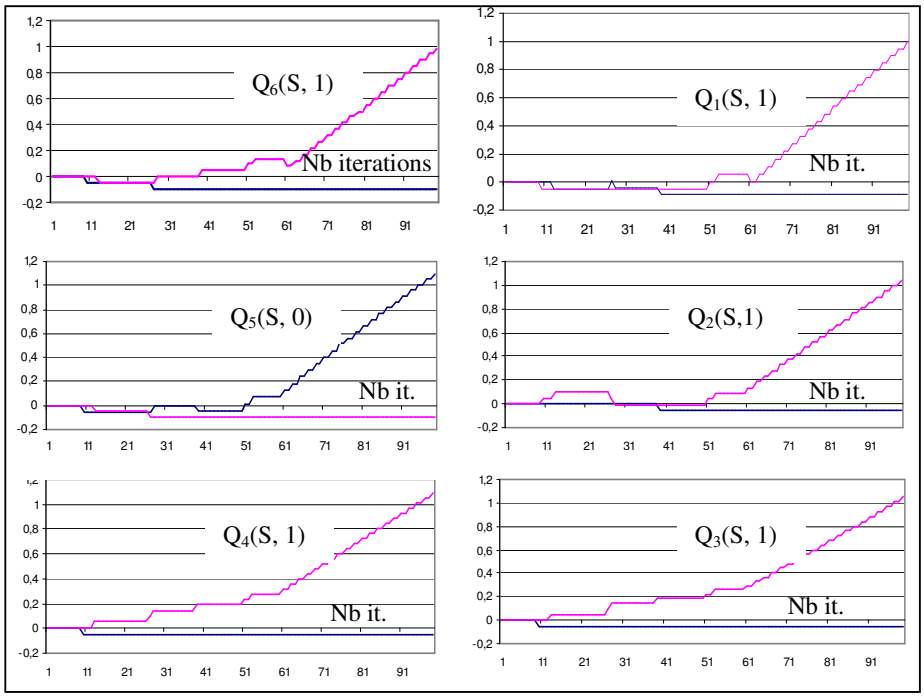

Figure 7. *Allure des fonctions valeurs Q (régime transitoire)*

5.3. Discussion

Nous avons comparé l'approche distribuée et l'approche centralisée. Dans le cas centralisé, un seul apprentissage est mené et les valeurs d'utilité Q(s, a) sont réunies dans une table unique de dimension 64*64. Dans ce cas, toutes les pattes reçoivent le même signal de renforcement selon les règles suivantes : une pénalité est envoyée à chaque patte si une des règles en vigueur du critique du Tableau 1 fournit une pénalité. Une récompense est envoyée à chaque patte si et seulement si, au moins une règle du critique fournit une récompense alors que les autres signaux de renforcement sont nuls.

Sur les simulations menées, l'apprentissage réalisé selon l'approche distribuée se révèle plus rapide et présente un nombre moyen de chutes par épisode plus faible. Une expérience typique montre un facteur 8 entre les deux approches en nombre d'itérations et un facteur 20 sur le nombre de chutes (Zennir *et al.*, 2003).

Les bons résultas obtenus en apprentissage peuvent être expliqués par le fait qu'il existe des équilibres de coordination (une marche périodique étant obtenue lorsque les acteurs maximisent leurs gains). Cependant, dans le cas centralisé, l'apprentissage est plus difficile (plus long et avec plus de chutes), car il y a beaucoup de pénalités pour peu de récompenses et la table des valeurs Q est plus grande (l'espace de recherche est plus vaste).

Dans cette expérimentation des objectifs individuels ont été fixés aux acteurs sur la base des règles de marche observées par les biologistes chez les insectes ; la marche apparaît comme un phénomène émergent résultant de la coordination des cycles locomoteurs des pattes, et un paramètre de contrôle tel que la durée de rétraction permet le changement d'allure. Dans cette simulation, l'approche centralisée avec fonction critique globale s'avère moins efficace.

6. Simulations de changement de trajectoire avec contrôle de la posture

6.1. Le contrôle de la posture

La posture est définie par la position et l'orientation relatives du corps par rapport aux pieds. Le contrôle de la posture consiste alors à déplacer le corps relativement aux pieds, afin d'améliorer, la stabilité et la mobilité du robot. Ainsi la posture de référence représentée en Figure 8 assure un bon compromis entre stabilité (centre de gravité du corps bas et placé à l'aplomb de l'isobarycentre du polygone de sustentation, polygone de sustentation plus grand que la projection verticale du corps sur le sol) et mobilité (facilité d'accès à l'ensemble des positions de l'espace atteignable par les pattes).

En reliant la position des pieds, qu'ils soient au sol ou non, on définit un polygone appelé, « polygone de configuration » qui, en fonction du terrain, n'est pas

nécessairement plan (Celaya *et al.*, 1998). Deux postures sont compatibles si elles admettent le même polygone de configuration.

Considérons un polygone de configuration quelconque P_Q, la posture optimale pour ce polygone de configuration est la posture qui lui est compatible et qui minimise la distance entre ce polygone de configuration P_Q et le polygone de configuration P_R défini par la posture de référence de la Figure 8. Ainsi la position du corps dans la posture optimale est celle qui réduit au minimum les distances entre les sommets respectifs des polygones P_R et P_Q.

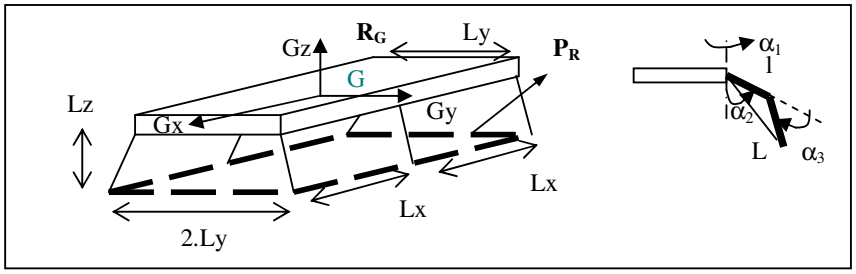

Figure 8. *Posture de référence et polygone de configuration P_R. Chaque patte est constituée de deux segments de longueur l et L et possède 3 degrés de libertés : ($-\pi/2<\alpha_1<\pi/2$, $0<\alpha_2<\pi$, $0<\alpha_3<\pi$)*

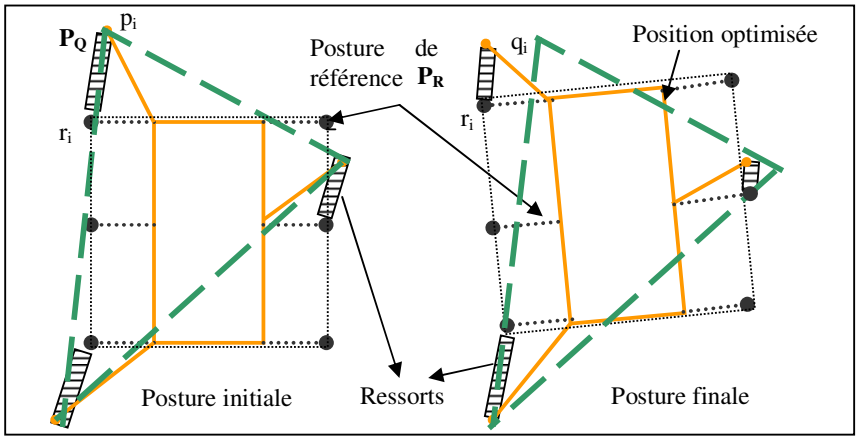

Figure 9. *Contrôle de la posture en utilisant l'analogie des ressorts. Dans ce cas particulier, le polygone de configuration est un triangle. En traits pleins sont représentés le corps du robot et les pattes dont les pieds restent en place sur le sol. Pour atteindre la posture finale le corps avance et pivote autour de Gz*

Dans notre modèle de posture, le polygone de configuration est défini par les seuls pieds qui sont en contact avec le sol et est alors confondu avec le polygone de sustentation, c'est-à-dire que nous admettons que les pattes qui sont en l'air n'ont pas d'influence sur la recherche d'une posture optimale.

Le contrôle de la posture consiste à atteindre la posture optimale en utilisant des mouvements coordonnés des pattes de sorte que le polygone de configuration reste inchangé (on dit que les mouvements sont conservatifs). Les effets du contrôle de la posture peuvent être illustrés par l'analogie suivante (Celaya *et al.*, 1998) : On suppose que les pieds en contact avec le sol sont solidement arrimés et que le polygone de configuration P_R correspondant à la posture de référence est fixé rigidement au corps (Figure 9).

Si chaque pied de la configuration courante est relié au sommet correspondant du polygone de référence par un ressort, alors le corps se déplace pour réduire au minimum l'énergie potentielle de telle sorte que les sommes des forces et couples exercés sur le corps s'annulent. La position finale du corps par rapport aux pieds correspond alors à la posture optimale pour le polygone de configuration P_Q. Le modèle de la posture est celui proposé par E. Celaya et J.M. Porta (Celaya *et al.*, 1998). Il est décrit par les expressions suivantes :

Soit le repère R_G lié à la posture de référence du robot comme indiqué en Figure 8. Dans le repère R_G, les positions q_i des pieds dans une posture Q se déduisent des positions p_i d'une posture compatible P par la transformation $T(\Omega)$ avec $\Omega = [x\ y\ z\ \phi\ \theta\ \psi]'$, où x,y,z sont des translations selon les axes du repère R_G et ϕ, θ, ψ sont des rotations autour des axes Gx, Gy et Gz. Nous avons alors la relation :

$$\begin{bmatrix} q_x^i \\ q_y^i \\ q_z^i \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & x \\ \sin(\phi) & \cos(\phi) & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\psi) & -\sin(\psi) & 0 \\ 0 & \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x^i \\ p_y^i \\ p_z^i \\ 1 \end{bmatrix} \quad \textbf{[3]}$$

La distance $D_{Q,R}(\Omega)$ entre la posture Q et la posture de référence R est :

$$D_{Q,R}(\Omega) = \sum_{i=1}^{6} \left\| q^i - r^i \right\|^2 = \sum_{i=1}^{6} (q_x{}^i - r_x{}^i)^2 + (q_y{}^i - r_y{}^i)^2 + (q_z{}^i - r_z{}^i)^2 \quad \textbf{[4]}$$

On en déduit le gradient par rapport à Ω de la distance dans la posture Q :

$$\left.\frac{\partial D_{Q,R}}{\partial x}\right|_{\Omega=0} = 2.\sum_{i=1}^{6}(p_x^i - r_x^i) \quad \left.\frac{\partial D_{Q,R}}{\partial \psi}\right|_{\Omega=0} = 2.\sum_{i=1}^{6}(r_y^i p_z^i - r_z^i p_y^i)$$
$$\left.\frac{\partial D_{Q,R}}{\partial y}\right|_{\Omega=0} = 2.\sum_{i=1}^{6}(p_y^i - r_y^i) \quad \left.\frac{\partial D_{Q,R}}{\partial \theta}\right|_{\Omega=0} = 2.\sum_{i=1}^{6}(r_z^i p_x^i - r_x^i p_z^i) \qquad \textbf{[5]}$$
$$\left.\frac{\partial D_{Q,R}}{\partial z}\right|_{\Omega=0} = 2.\sum_{i=1}^{6}(p_z^i - r_z^i) \quad \left.\frac{\partial D_{Q,R}}{\partial \phi}\right|_{\Omega=0} = 2.\sum_{i=1}^{6}(r_x^i p_y^i - r_y^i p_x^i)$$

Pour exécuter un mouvement conservatif de la posture Q vers la posture R, les mouvements doivent être coordonnés de telle sorte que le polygone de configuration ne soit pas déformé pendant le processus. Chaque patte est constituée de deux segments de longueur l et L et possède 3 degrés de libertés (Figure 8). E. Celaya et J.M. Porta proposent d'implémenter le mécanisme de contrôle de la posture selon six processus d'ajustement séparés, chacun d'entre eux correspondant à un degré de liberté du corps. Ainsi lorsqu'une composante du gradient est différente de zéro, l'ajustement correspondant est chargé de réaliser un petit déplacement de tous les pieds selon le degré de liberté correspondant pour diminuer la distance $D_{Q,R}$.

Puisque les ajustements procèdent par succession de petits déplacements, ils sont assimilés à des mouvements effectués en ligne droite dans les directions données par le vecteur des dérivées partielles de q^i par rapport aux six paramètres. Ces vecteurs s'expriment par :

$$\left.\frac{\partial q^i}{\partial x}\right|_{\Omega=0} = \begin{bmatrix}1\\0\\0\end{bmatrix} \quad \left.\frac{\partial q^i}{\partial \psi}\right|_{\Omega=0} = \begin{bmatrix}1\\-p_z^i\\p_y^i\end{bmatrix}$$
$$\left.\frac{\partial q^i}{\partial y}\right|_{\Omega=0} = \begin{bmatrix}0\\1\\0\end{bmatrix} \quad \left.\frac{\partial q^i}{\partial \theta}\right|_{\Omega=0} = \begin{bmatrix}p_z^i\\0\\-p_x^i\end{bmatrix} \qquad \textbf{[6]}$$
$$\left.\frac{\partial q^i}{\partial z}\right|_{\Omega=0} = \begin{bmatrix}0\\0\\1\end{bmatrix} \quad \left.\frac{\partial q^i}{\partial \phi}\right|_{\Omega=0} = \begin{bmatrix}-p_y^i\\p_x^i\\0\end{bmatrix}$$

Ainsi chaque ajustement exécute des petits déplacements des sommets du polygone de configuration lié à la posture de référence dans la direction correspondant à l'un de ces vecteurs et réalise ainsi une descente de gradient selon un degré de liberté donné. Connaissant les positions des articulations des épaules et la position des pieds, les angles α_i des segments de patte (Figure 8) sont calculés à chaque ajustement. Lorsque le gradient est proche de zéro, le corps se trouve dans la

posture optimale, et il a donc changé de position et d'orientation par rapport au repère R_G et aussi par rapport à un repère absolu lié à l'environnement.

6.2. Description du changement de trajectoire avec contrôle de la posture

Nous utilisons le modèle géométrique de la posture décrit en section 6.1 pour simuler l'apprentissage de changements de trajectoire de façon à ce que, partant d'une position et orientation initiale quelconque, le robot rattrape une trajectoire proche et parallèle à l'axe Ox dans un repère lié au terrain et telle que l'axe principal du robot soit pratiquement parallèle à l'axe Ox. Cette trajectoire à atteindre est définie par l'écart maximal toléré Δy_f avec l'axe Ox et la rotation maximale $\Delta\theta_f$ tolérée autour de l'axe Gz (Figure 10).

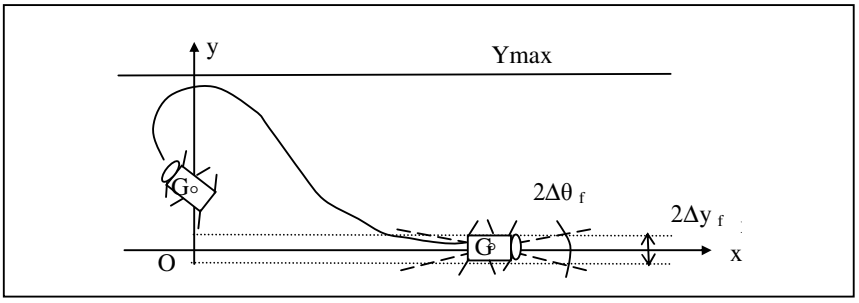

Figure 10. *Contrôle de changement de trajectoire. Gz axe vertical passe par le centre de gravité du robot*

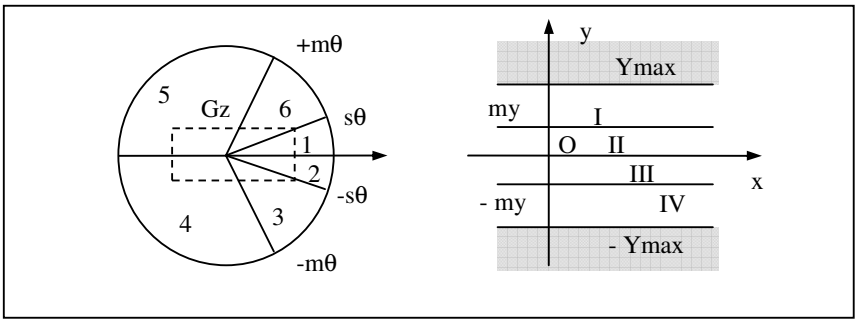

Figure 11. *Codage de l'espace des états*

Pour tourner, les insectes peuvent conserver le même type de marche qu'en déplacement rectiligne mais en modifiant l'amplitude des pas (Cruse, 1990). Nous supposons alors dans la simulation que le robot évolue selon la marche tripode (la plus rapide et la plus fréquente des marches chez les insectes) et que chaque patte peut pousser vers l'avant ou vers l'arrière indépendamment des autres avec une amplitude fixe. Ainsi, le robot peut marcher en avant ou en arrière et peut tourner.

L'apprentissage consiste alors à choisir les amplitudes des pas depuis tout état de l'espace, pour que le robot finisse par atteindre son objectif. Pour décrire la position et l'orientation de l'hexapode, quatre zones parallèles à l'axe Oy et six secteurs autour de l'axe Gz sont définis comme représenté en Figure 11. Ainsi le nombre d'états possibles est |S| = 24.

Les actions sont décrites par l'amplitude algébrique des mouvements de balancement des pattes entre la position de posture de référence et la prochaine position de contact au sol. L'amplitude de tels mouvements peut prendre les valeurs -avx ou +avx. Les pattes sont divisées en deux groupes, chacun des groupes étant constitué des trois pattes qui soutiennent alternativement le robot pendant la marche tripode. Chaque acteur (un par patte) actualise sa propre table de valeur de Q de dimension $24*2^3$ (24 états, et 8 actions possibles des 3 acteurs du même groupe).

Le signal de renforcement est calculé selon le critique suivant :

- une récompense est attribuée au groupe si les objectifs en position et en rotation sont atteints et si la variation de x_G est positive,
- une pénalité si la distance de l'axe Ox est plus grande que my,
- un signal de renforcement nul dans les autres cas. La procédure de simulation est alors la suivante :

Initialiser les tables de valeur Q à 0

Debut : Répéter jusqu'à ce que la série de tests soit validée

Choisir la position initiale du centre de gravité (x_G, y_G, z_G) et la rotation initiale (θ_G) autour de l'axe Gz du robot placé dans la posture de référence :

x_G=0, z_G =0, y_G valeur aléatoire entre - Ymax + Ymax,

θ_G valeur aléatoire entre $-\pi$ et π.

Dans chaque groupe appliquer l'algorithme Q-multiacteur de la Figure 5

Si des récompenses en nombre suffisant sont reçues ou si le nombre de pas est plus grand qu'une limite prédéfinie, allez à 'Début'

Figure 12. *Procédure de la simulation pour l'apprentissage d'un changement de trajectoire avec contrôle de la posture*

La série de tests consiste à atteindre et suivre suffisamment longtemps la trajectoire fixée comme objectif à partir de positions et d'orientations différentes.

6.3. Résultats

Les paramètres géométriques du robot et de l'environnement sont : Lx =1, Ly = 0.2, Lz = 0.5 (voir Figure 8), les deux segments constituant une patte ont la même longueur égale à 0.35; avx = 0.25, sθ = 0.1 rd, mθ = 1.25 rd, my = 0.2, Ymax = 2., Δy_f = 0.2, $\Delta\theta_f$ = 0.1 rd. Les longueurs et distances sont des grandeurs réduites (l'unité de longueur est Rx). Les positions et orientations initiales pour les tests sont : x_G=0, z_G=0, y_G et θ_G prenant respectivement pour les huit trajectoires de test les valeurs suivantes : {(0,0), (0.75, π/4), (0.5, π/2), (-1.5, 3π/4), (0, π), (1.5, -3π/4), (-0.5, -π/2), (-0.75, -π/4)}.

Les paramètres d'apprentissage sont les suivants : α = 0.05, γ= 0.9, ε = 0.01; la durée d'une boucle d'apprentissage avant test est au maximum de 1000 épisodes de 300 itérations. Une nouvelle position de départ est tirée aléatoirement avant chaque série de 300 itérations. Si 50 récompenses successives sont reçues lors d'un test, la trajectoire désirée est supposée atteinte. L'allure des trajectoires obtenues varie d'une expérience à l'autre car plusieurs solutions sont possibles. Nous présentons en Figure 13 des trajectoires de test obtenues pour trois des positions et orientations initiales.

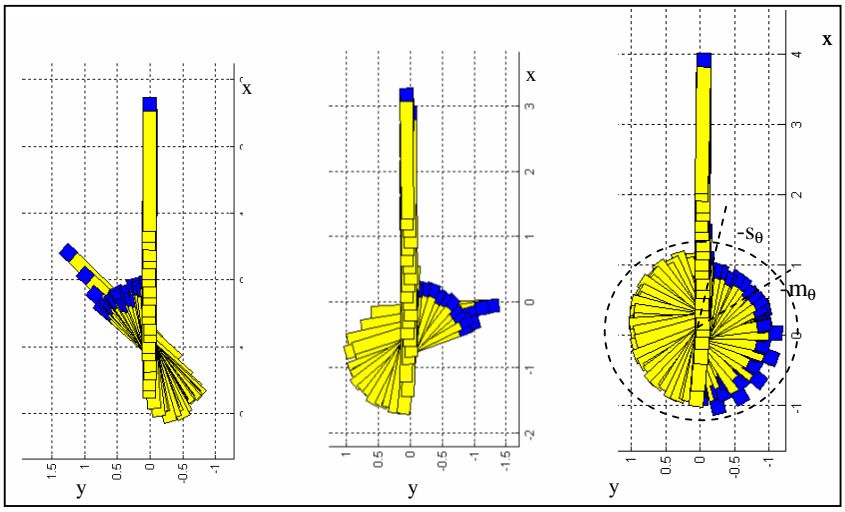

Figure 13. *Exemples de trajectoires apprises par le robot hexapode depuis les états initiaux* (0.75, π/4), (-0.5, -π/2), (0, π) *: trace au cours du temps dans le plan Oxy.*

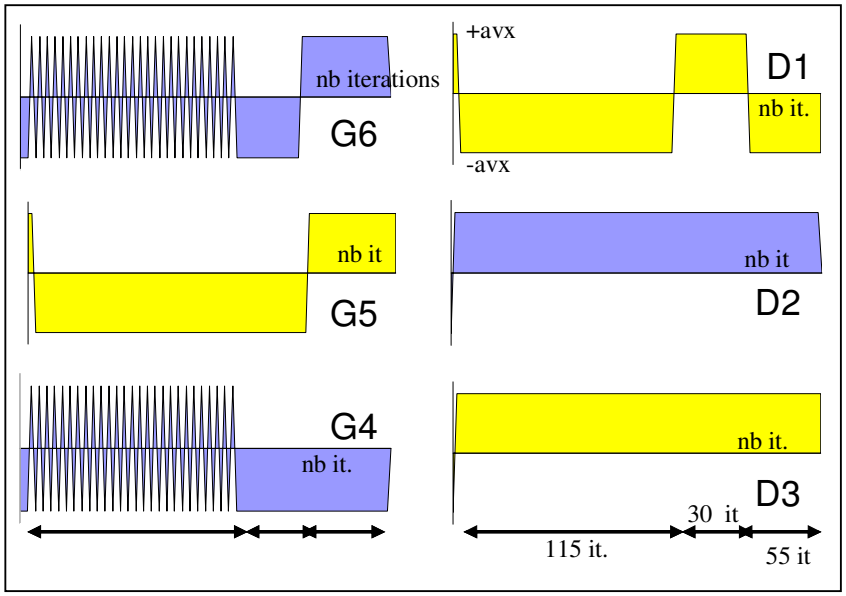

Figure 14. *Amplitude des mouvements lors du demi tour*

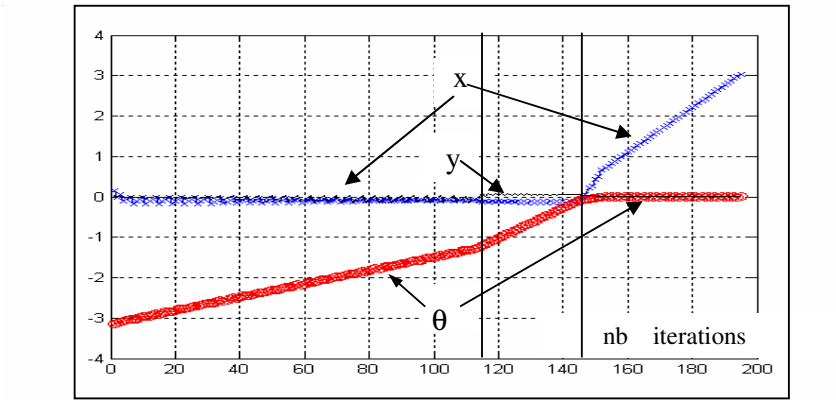

Figure 15. *Position de G et orientation autour de Gz lors du demi tour*

En Figure 14 sont visualisées les amplitudes des mouvements appris par chaque contrôleur lors du demi-tour. Trois phases sont clairement identifiables correspondant au trois secteurs angulaires (notés 4, 3 et 2 sur la Figure 11) visités

lors du demi-tour de la Figure 13. A chaque itération les groupes alternent entre proaction et rétraction. On peut noter qu'en fin de trajectoire les mouvements des pattes dans chaque groupe {D1, D3, G5} et {D2, G4, G6} ne sont pas de même signe, ce qui correspond à une trajectoire légèrement oscillante autour de la trajectoire y = 0 (voir exemple de la Figure 9 pour le groupe {D2, G4, G6}). Il est normal de trouver ce type de trajectoire, car la fonction de renforcement l'accepte tant que l'angle de rotation est compris entre ± $\Delta\theta_r$. En Figure 15 sont représentées les évolutions de la position du centre de gravité et l'orientation autour de Gz lors du demi tour. Elles sont conformes à ce qui est attendu : on constate que le robot atteint la trajectoire désirée tout d'abord en tournant puis en se translatant selon l'axe Ox tout en restant au voisinage de la droite y=0.

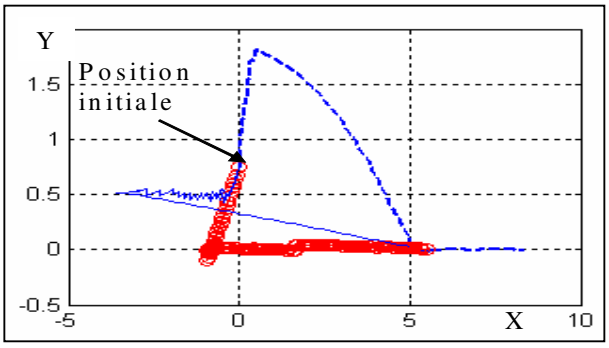

Figure 16. *Trajectoires obtenues dans le plan Oxy pour trois valeurs différentes de my. Lorsque my (seuil sur l'axe y) augmente, les trajectoires s'écartent de l'axe Ox*

Nous visualisons sur la Figure 16, l'influence d'une augmentation de la valeur my sur l'allure de la trajectoire apprise, les autres paramètres restant inchangés. Quand my croît, la trajectoire suivie par le robot peut s'écarter plus largement de l'axe Ox , alors que comme dans le cas de la Figures 15, lorsque my est plus petit, on contraint le robot à tourner sur place car il est très vite pénalisé s'il s'écarte de l'axe Ox.

Notons aussi que les trajectoires suivies par le robot pour des points de départ symétriques par rapport à l'axe Ox ne présentent pas forcément elles-mêmes de symétrie par rapport à cet axe. En effet, la série de test peut-être passée avec succès alors que les politiques apprises sont sous optimales, d'autre part dans la zone où le robot reçoit un signal de renforcement nul, le robot peut adopter des trajectoires différentes.

6.4. Comparaison des stratégies individuelle et collective

Nous avons comparé la stratégie individuelle pour laquelle les acteurs (les contrôleurs de mouvement de chaque patte) apprennent en ignorant l'existence des autres acteurs, chacun actualisant une table Q de dimension 24*2, et une stratégie collective pour laquelle les acteurs dans chacun des deux groupes de trois prennent en compte l'existence des autres et actualisent les 24*8 valeurs comme décrit dans l'algorithme Q-multiacteur (Figure 5). Une différence importante avec l'étude menée en section 5 est que le critique n'est pas capable dans cette expérience, d'attribuer un signal de renforcement individuel (le signal de renforcement est le même pour tous les membres du groupe).

La comparaison porte sur le nombre d'épisodes d'apprentissage nécessaires pour passer avec succès la série de tests en fonction de ε, probabilité pour chacun des acteurs de choisir aléatoirement le mouvement à exécuter plutôt que d'appliquer l'action la plus profitable. Pour chaque variante de l'algorithme, les valeurs aléatoires des positions initiales sont lues dans le même ordre à partir d'un même fichier. Les autres paramètres de simulation sont identiques. Les résultats sont portés sur le graphe de la Figure 17.

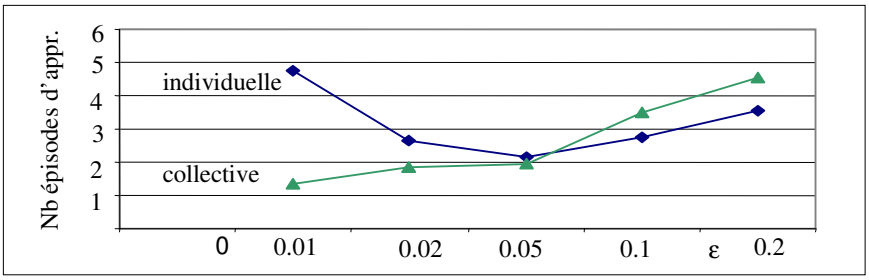

Figure 17. *Comparaison entre stratégies individuelle et collective. La stratégie collective (algorithme Q-multiacteur avec deux groupes) permet d'apprendre plus vite pour un taux d'exploration faible*

L'algorithme Q-multiacteur est plus efficace (moins de passages dans la boucle d'apprentissage sont nécessaires pour passer avec succès la série des huit tests) pour les probabilités d'exploration les plus faibles. Ce constat est confirmé sur une expérience menée avec une valeur ε nulle où nous constatons que, sur 10 essais d'apprentissage (limitées à 10 boucles d'apprentissage), dans le cas où les acteurs

adoptent une stratégie individuelle, seulement 11% des tests sont positifs (et la série des huit tests validant l'apprentissage échoue à chaque expérience), alors que 57% sont positifs dans le cas où l'actualisation est dépendante des autres acteurs (et 3 apprentissages sur les 10 ont réussi).

Le signal de renforcement étant identique pour toutes les pattes qui sont au sol (contrairement à l'étude menée en section 5, le critique n'est pas capable de déceler quelle action individuelle est correcte ou incorrecte), il est préférable dans ce problème particulier de tenir compte des actions des autres membres du groupe lorsque l'exploitation est prédominante. Lorsque l'exploration devient plus importante ($\varepsilon > 0.05$), le nombre d'épisodes nécessaires à l'apprentissage croît mais, dans cette expérience, la stratégie collective ne permet pas de réduire le nombre d'épisodes nécessaires à l'apprentissage.

7. Conclusion

La mise en œuvre d'un apprentissage par renforcement dans un environnement distribué dépend du modèle relationnel entre les acteurs contribuant à une même tâche. Dans une architecture centralisée, l'information converge vers une entité unique chargée de décider des actions de tous. L'apprentissage par renforcement est mené comme s'il s'agissait d'un seul acteur dont les actions possibles sont les actions conjointes. Cette approche a l'avantage d'offrir une vision globale du problème de coordination entre les acteurs, mais ne permet pas de tenir compte du contexte local des acteurs (en particulier un signal de renforcement unique pénalise ou récompense uniformément les acteurs qu'ils contribuent ou non individuellement au succès ou à l'échec de la tâche). L'espace de recherche des couples (état, action) croît de façon exponentielle avec le nombre d'acteurs.

Dans un modèle distribué, les acteurs mènent leur propre apprentissage à partir d'informations d'état et d'un signal de renforcement qui peuvent leur être spécifiques. L'espace de recherche des couples (état, action) croît de façon proportionnelle avec le nombre d'acteurs. Se posent alors les problèmes de la définition des objectifs locaux compatibles avec l'objectif global et de la coordination entre acteurs. Selon une stratégie individuelle, des objectifs locaux sont fixés de telle sorte que les acteurs mènent leur apprentissage en s'ignorant. L'exemple de l'apprentissage de marches périodiques d'un robot hexapode mené selon un algorithme Q-learning distribué et une stratégie individuelle en est une illustration, l'approche distribuée se révélant, sur cet exemple, plus efficace que l'approche centralisée. Selon une stratégie collective, les acteurs observent les autres pour améliorer leur politique. Nous avons testé un mode de coordination, dans lequel un acteur tient compte a posteriori des actions exécutées par les acteurs d'un même groupe pour actualiser l'utilité de ses actions. Dans le cadre d'une simulation de changement de trajectoire avec contrôle de la posture, l'algorithme testé avec deux groupes s'avère plus performant que l'algorithme Q-learning distribué avec

stratégie individuelle pour apprendre au robot hexapode à rejoindre une trajectoire prédéfinie.

Des travaux futurs permettront d'étudier d'autres stratégies collectives basées sur la coopération. L'apprentissage de la marche en terrain non plat, l'évitement d'obstacles et la résolution de problèmes de navigation sont des sujets potentiels d'application intéressants pour explorer ces stratégies.

8. Bibliographie

Barto A. G. and Anandan P. «Pattern Recognition Stochastic Learning Automata». *IEEE Transaction on System, Man and Cybernetics*. SMC-15, 1985, pp. 360-375.

Brooks R. A. «A robot that walks; emergent behaviors from a carefully evolved network». *Neural computation*, 1989, pp.365-382.

Buffet O. Une double approche modulaire de l'apprentissage par renforcement pour des agents intelligents adaptatifs. Thèse en informatique, Equipe MAIA, Université Henri Poincarré, Nancy 1, 2003, 213 p.

Claus C. and Boutillier C. «The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems». *AAAI*, 1998, pp.746-752.

Celaya E. and Porta J.M. «A Control Structure for the locomotion of a legged Robot on difficult Terrain». *IEEE Robotics and Automation Magazine*, vol. 5, n°. 2, 1998, pp.43-51.

Cruse H. «The control of body position in the stick insect (carausius morosus), when walking over uneven terrain». *Biological cybernetics*, vol. 24, 1976, pp.25-33.

Cruse H. «What mechanisms coordinate leg movement in walking arthropods?». *Trends in neurosciences*, vol.13, 1990, pp.15-21.

Dean J. «A model of leg coordination in the stick insect, carausius morosus : I.A geometrical consideration of contralateral and ipsilateral coordination mechanisms between two adjacent legs». *Biological cybernetics*, vol. 64, 1991, pp.393-402.

Graham D. «Pattern and control of walking in insects». *Advances in insect physiology*, vol.18, 1985, pp.31-140.

Hu J. and Wellman M. P. «Multiagent Reinforcement Learning: Theoretical Framework and an Algorithm». *15th International Conference on Machine Learning*, in Madison, Wisconsin, USA, 1998, pp.242-250.

Johannet A. and Sarda I. «Gait learning of hexapod robot with neural networks : from simulation to realization». *2ème congrés Mécatroique Franco-Japonais de Takamatsu*, 1994, 4 p.

Kingsley D. A. , Quinn R. D. and Ritzmann R. E. «A cockroach inspired robot with artificial muscles». *International Symposium on Adaptive Motion of Animals and Machines*, Kyoto, Japan, 2003, 7 p.

Kirchner F. «Q-learning of complex behaviours on a six-legged walking machine». *Robotics and autonomous systems*, 1998, pp.253-262.

Littman, M. L. «Value-function reinforcement learning in Markov games». *Journal of cognitive Systems Research 2*, Elsevier, 2001, pp. 55-66.

Porta J.M. ρ-Learning: A robotics oriented reinforcement learning algorithm. Technical Report Instituto de Robótica e Informática Industrial (UPC-CSIC), Barcelona, Spain, 2000, 14 p.

Randall M.J. Stable adaptive neural control of systems with closed kinematic chains applied to biologically-inspired walking robots. These in philosophy. Bristol : Faculty of engineering, University of the West of England, Bristol, 1999, 322 p.

Sehad S. Contribution à l'étude et au développement de modèles connexionnistes à apprentissage par renforcement : application à l'acquisition de comportements adaptatifs. Thèse de génie informatique et traitement du signal. Montpellier : Université de Montpellier II, 1996, 112 p.

Svinin M.M., Yamada K. and Ueda K. «Emergent synthesis of motion patterns for locomotion robots». *Artificial intelligence in engineering,* 2001, pp.353-363.

Sutton R.S. and Barto A.G. *Reinforcement Learning*. Mit press, Cambridge, Bradford book, 1998, 322 p. ISBN 0-262-19398-1.

Touzet C. et Sarzeaud O. «Application d'un algorithme d'apprentissage par pénalité /récompense a la géneration des formes locomotrices hexapodes». *Journées de rochebrune, AFCET IA et cognition*, 1992, 5 p.

Touzet C. «Distributed Lazy Q-learning for Cooperative Mobile Robots». *International Journal of Advanced Robotic Systems*, vol. 1, No. 1, January 2004, pp. 5-13.

Tumer K. and Wolpert D. «Collective Intelligence and Braess'Paradox». *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, Austin, TX, August 2000. pp. 104-109.

Wang X. and Sandholm T. «Reinforcement Learning to Play an Optimal Nash Equlibrium in Team Markov Games». Becker S., Thrun S., Obermayer K., Eds., *Advances in Neural Information Processing Systems 15*, MIT Press, Cambridge,MA, 2003, p. 1571-1578.

Watkins C.J.C.H. Learning from Delayed Rewards, Ph.D. thesis, Cambridge University, 1989, 234p.

Zennir Y., Couturier P. and Bétemps M. «Emergence of the gaits of a hexapod robot using distributed reinforcement learning». *Proceeding of the IASTED International conference on Intelligent systems and control*, Salzburg, Austria, 2003, pp.106-111.

Reçu le : 29 Octobre 2004

Accepté le : 20 septembre 2005 après révision.