International Conference on Applied Informatics ICAI09, November 15-17, 2009, B.B.Arréridj, Algeria

Evitement d'Obstacles pour un Robot Mobile en Utilisant L'apprentissage par Renforcement

L. Cherroun, M. Boumehraz
Département d'Automatique, Université de Biskra - Algérie
E-mail: cherroun_lakh@yahoo.fr medboumehraz@netcourrier.com

Résumé

La programmation d'un robot mobile est une tâche importante nécessitant une modélisation complète de l'environnement. Dans ce travail, on présente une technique de navigation intelligente d'un robot mobile qui nécessite seulement un signal scalaire comme information de retour indiquant la qualité de l'action appliquée. Au lieu de programmer un robot pour qu'il effectue une mission, nous allons le laisser apprendre seul sa propre stratégie. L'algorithme Q-learning de l'apprentissage par renforcement est utilisé pour la navigation d'un robot mobile en discrétisant les espaces d'états et d'actions. Afin d'améliorer les performances, une extension de cet algorithme aux cas continus consiste à introduire des connaissances à priori par un système d'inférence flou est a utiliser ensuite l'algorithme de Q-learning flou pour la tâche de navigation d'un robot mobile.

Mots clés : robot mobile, navigation intelligente, apprentissage par renforcement, Q-learning, système d'inférence flou.

1. Introduction

L'une des missions de base dans une tâche de navigation d'un robot mobile est la capacité d'évitement d'obstacles. C'est une tâche importante que doivent posséder tous les robots mobiles, puisque ceci permet au robot de se déplacer dans un environnement inconnu sans collisions. Il est possible d'utiliser un paradigme réactif qui décrit le lien entre la perception et l'action pour accomplir les objectifs du robot [1]. Une stratégie d'évitement d'obstacles avec une capacité d'apprentissage peut être réalisée en utilisant l'apprentissage par renforcement; ou le robot reçoit seulement un signal scalaire comme information de retour. Le signal de renforcement permet au navigateur d'ajuster sa stratégie pour améliorer ses performances. C'est une modification automatique du comportement du robot dans son environnement de navigation comme réalisé dans [2][3].

L'apprentissage par renforcement est une méthode de commande optimale, parce qu'on part d'une solution inefficace que l'on améliore progressivement en fonction de l'expérience acquise pour résoudre un problème de décision séquentielle [4][5].

Pour utiliser l'apprentissage par renforcement, plusieurs approches sont possibles. La première consiste à discrétiser manuellement le problème afin de fournir des espaces d'états et d'actions qui pourront être utilisés directement par des algorithmes utilisant des tableaux de Q. Il faut cependant faire attention au choix des discrétisations afin qu'elles permettent un apprentissage correct en fournissant des états et des actions qui contiennent notamment des récompenses cohérentes [5][6]. Ce choix peut être relativement simple pour des capteurs de distance, mais complexe voir impossible si l'espace d'entrée est plus abstrait ou si sa structure est un peu connu.

La seconde méthode consiste à travailler directement dans les espaces d'états et d'actions continus en utilisant des méthodes d'approximation des fonctions. En effet, pour utiliser l'apprentissage par renforcement, il est nécessaire d'estimer correctement la fonction de qualité Q [7]. Cette estimation peut se faire directement par un approximateur de fonction continu comme les réseaux de neurones ou les systèmes d'inférence floue [6][8][9][10]. L'utilisation de ces approximateurs permet de travailler directement dans l'espace continu et de limiter les effets de parasites qui pourraient apparaitre suite à un mouvais choix de discrétisation [11][12][13].

Ce papier est organisé comme suit : dans la section 2, un bref exposé sur la méthode d'apprentissage par renforcement sera présenté. La section 3 sera consacrée à l'application de l'algorithme Q-learning pour une tâche d'évitement d'obstacles. Mais l'impossibilité d'exécution des actions discrètes a conduit à l'extension de cet algorithme aux espaces d'états et d'actions continus et son application à l'évitement d'obstacles (section 4). Cela permet d'améliorer les performances du robot mobile.

2. L'apprentissage par renforcement

L'apprentissage par renforcement est une méthode d'apprentissage à partir de l'expérience afin de trouver, par un processus essais-erreurs, l'action optimale à effectuer pour chacune des situations que l'agent va percevoir pour maximiser ses récompenses. L'idée fondamentale de l'apprentissage par renforcement est d'améliorer une politique courante après chaque interaction avec l'environnement [5][14].

A l'instant t l'agent (le robot dans notre cas) perçoit son environnement par la perception s_t, et peut agir en exécutant l'action a_t, et d'autre part ; reçoit une récompense (r_t).

Le but de l'agent dans le cadre de l'apprentissage par renforcement est de trouver le comportement le plus efficace pour maximiser l'espérance de ses gains à la suite de chaque transition (s_t, a_t, r_t, s_{t+1}) [4][5][14], où s_{t+1} est la situation suivante.

Le comportement est défini par une *politique* $\pi : \{S, A\} \rightarrow [0,1]$, notée $\pi(s) = a$. Le but de l'agent est de trouver la politique optimale π^* maximisant la récompense à long terme en utilisant la fonction de valeur comme mesure de performance [5]. Généralement la fonction de valeur est définit dans un problème de la forme d'un processus de décision markovien PDM par :

$$V_\pi(s) = E_\pi(R_t | s_t = s) = E_\pi\left(\sum_{k=1}^{\infty} \gamma^k r_{t+k} | s_t = s\right) \quad (1)$$

Ou $\gamma \in [0,1[$ est un facteur de décroissement qui permet de régler l'importance que l'on donne aux retours futurs par rapport aux retours immédiats.

La plus part des algorithmes d'apprentissage par renforcement utilisent une fonction de qualité représentant la valeur de chaque paire état-action pour obtenir un comportement optimale [5][16]. Elle donne pour chaque état, le retour futur si l'on suit cette politique π :

$$Q^\pi(s,a) = E_\pi(R_t | s_t = s, a_t = a) \quad (2)$$

La qualité optimale est :

$$Q^*(s,a) = \max_\pi Q^\pi(s,a) \quad (3)$$

On obtient alors :

$$Q^*(s,a) = E(r_{t+1} + \gamma V^*(s_{t+1}) | s_t = s, a_t = a) \quad (4)$$

L'apprentissage par différences temporelles (*TD*) est une combinaison des méthodes de Monte Carlo et des méthodes de programmation dynamique. Ces méthodes permettent d'apprendre directement sans avoir un modèle de l'environnement en évaluant l'action sans avoir besoin d'arriver au but final (*bootstrap*) [5].

Le Q-learning est l'algorithme de différences temporelles le plus utilisé. Il permet d'apprendre la fonction de qualité en interagissant avec l'environnement [14][16], en mettant à jour itérativement la fonction courante $Q^\pi(s_t, a_t)$ à la suite de chaque transition (s_t, a_t, r_t, s_{t+1}). Cette mise à jour se fait sur la base de l'observation des transitions instantanées et de leurs récompenses associées par l'équation suivante :

$$Q(s_t,a_t) \leftarrow Q^\pi(s_t,a_t) + \alpha\left[r_t + \gamma \max_{a \in A(s)} Q(s_{t+1},a_t) - Q(s_t,a_t)\right] \quad (5)$$

Où $\alpha \in [0,1]$ est un cœfficient d'apprentissage qui doit diminuer pour tendre vers 0.

Les fonctions de qualité sont stockées sous la forme de tableaux : une ligne correspond aux qualités des différentes actions pour un état donné. Au début lorsque la table ne contient pas suffisamment de données, une composante aléatoire est ajoutée de façon à ne pas restreindre les actions éligibles au petit nombre des actions déjà essayées. Au fur et à mesure que la table se remplit, cette composante aléatoire est réduite afin de permettre l'exploitation des informations reçues et d'obtenir une bonne performance.

3. Application de Q-learning pour des espaces d'états et d'actions discret

A chaque étape le robot doit définir, l'état dans lequel il se trouve et à partir de cet état, il doit prendre une décision sur l'action à exécuter. En fonction du résultat obtenu lors de l'exécution de cette action, il est soit puni, pour diminuer la probabilité d'exécution de la même action dans le futur, soit récompensé, pour favoriser ce comportement dans les situations pareilles.

Pour une tâche d'évitement d'obstacles, le robot utilise ses capteurs pour observer l'espace autour de lui. On a limité la zone de vision à 90°. Cette limitation a approché le comportement et les capacités du robot simulé à ceux des robots réels, qui possèdent par exemple des capteurs ultrasonores ou infrarouge.

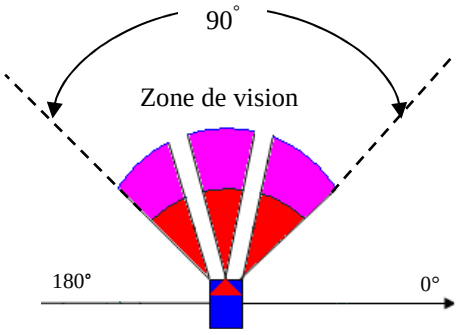


Figure 1. *Le modèle de vision utilisé*

A chaque pas de robot, les capteurs délivrent une mesure de la distance des obstacles proches. On a divisé l'espace en 3 secteurs. Chaque secteur est divisé aussi en 3 zones : la zone critique, la zone proche et la zone éloignée (figure 2). Il faut mentionner, qu'on a réuni tous les états dans lesquels un obstacle se trouve.

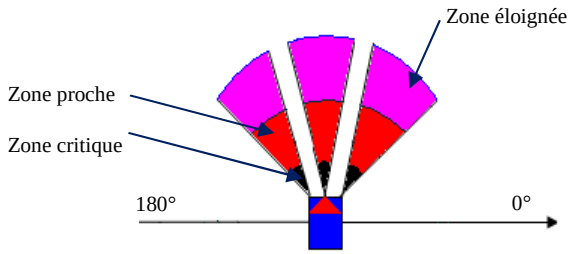


Figure 2. *Définition des états.*

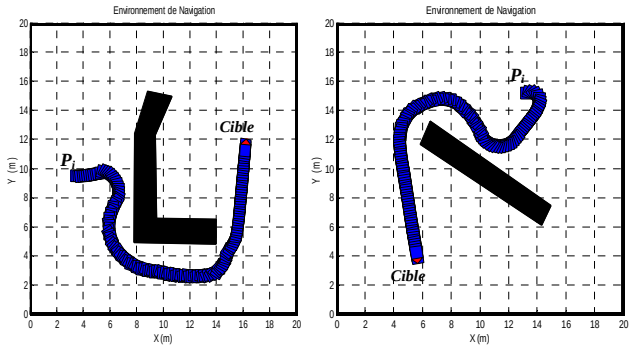
Les actions utilisées sont : avancer, tourner à droite et tourner à gauche.

Pendant le déplacement, le robot reçoit les renforcements suivants :

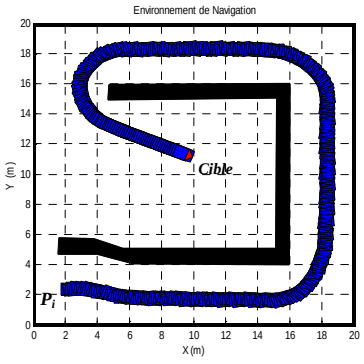
$$r = \begin{cases} -4, & \text{Si le robot percute un obstacle,} \\ -2, & \text{Si } d_i \text{ diminue et } d_i < \frac{l_c}{2}, \\ -1, & \text{Si } d_i < l_c \text{ diminue et } d_i > \frac{l_c}{2}, \\ 0, & \text{ailleurs,} \end{cases}$$

où l_c est la distance maximale de vision du capteur.

Les figures 3 (a-b-c) illustrent les trajectoires du robot mobile après la phase d'apprentissage.



-a- **- b-**



-c-

Figure 3 (a-b-c). *Navigation avec évitement d'obstacles*

Lors de l'apprentissage, on observe la maximisation des renforcements moyens ce qui montre l'amélioration du comportement du robot au cours du temps comme présenté sur la figure suivante :

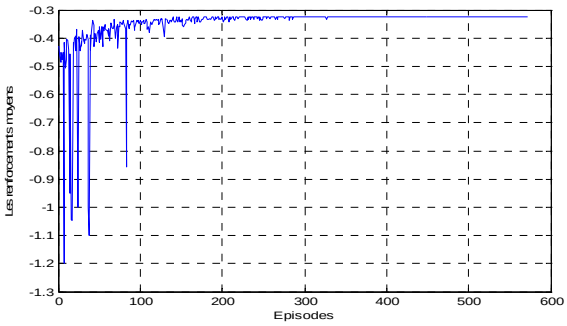


Figure 4. *Les valeurs moyennes du renforcement*

Dans cette implémentation, les espaces utilisés sont discrets, ce qui limite l'application réelle de cet algorithme. Comme extension de cet algorithme aux cas continu, l'utilisation des systèmes d'inférence floue est une solution possible. Dans le but d'améliorer les performances du robot mobile, on a utilisé une stratégie à base d'un système d'inférence floue caractérisée par l'introduction des connaissances disponibles à priori pour que le comportement initial soit acceptable.

4. Généralisation de Q-learning en utilisant la logique floue

L'utilisation des algorithmes d'apprentissage par renforcement exige le stockage des valeurs de la fonction de qualité pour tous les couples (état, action). Dans les problèmes discrets de faible dimension ; on peut utiliser des tableaux. Mais dans le cas des espaces d'états et d'actions continus comme la tâche de navigation d'un robot mobile ; le nombre de situations est infini et la représentation de la fonction Q par des tableaux est impossible. Les approximateurs universels, comme les réseaux de neurones et les systèmes d'inférence floue offrent des solutions prometteuses pour l'approximation des valeurs d'utilité [8][17].

L'utilisation des systèmes d'inférence floue est une solution prometteuse [15][18][19]. La tâche consiste à approcher la fonction de qualité Q par la fonction :

$$s \rightarrow y = \hat{Q} = SIF\ (s) \qquad (6)$$

Le principe de cette extension naturelle est de proposer plusieurs conclusions pour chaque règle (prémisse) et à associer à chaque conclusion une fonction de qualité qui sera évaluée incrémentalement au cour du temps. Le processus d'apprentissage permet alors de déterminer l'ensemble des règles maximisant les renforcements futurs [9][10][18]. Cette version floue de l'algorithme Q-learning est appelée le Q-Learning flou.

La base des règles initiale en utilisant un modèle flou de type Takagi-Sugeno d'ordre 0 est composée de m règles et *N* conclusions proposées de la forme [9][18]:

$$\begin{aligned} &\text{Si } s \text{ est } S_i \text{ Alors} && y = a[i,1] \text{ avec } q[i,1] = 0 \\ &\text{ou} && y = a[i,2] \text{ avec } q[i,2] = 0 \\ &&& \dots \\ &\text{ou} && y = a[i,N] \text{ avec } q[i,N] = 0 \end{aligned} \quad (7)$$

Ou $q(i,j)$ avec $i = 1...m$ et $j = 1...N$, sont des solutions potentielles dont la valeur est initialisée à 0. Durant l'apprentissage, la conclusion de chaque règle est choisie au moyen d'une politique d'exploration-exploitation notée (*PEE*) ou $PEE(i) \in \{1...N\}$. Dans ce cas, la sortie inférée est donnée par :

$$A(s) = \sum_{i=1}^{m} w_i(s).a[i, PEE(i)] \quad (8)$$

Et la qualité de cette action sera :

$$\hat{Q}(s, A(s)) = \sum_{i=1}^{N} w_i(s).q[i, PEE(i)] \quad (9)$$

4.1 Application de Q-learning flou pour la navigation d'un robot mobile

L'application se résume en l'implémentation d'un contrôleur flou pour la navigation d'un robot mobile. Sa base de règles est améliorée en ligne en utilisant un signal de renforcement.

Le système de navigation utilisé a comme variables :

- Les entrées sont les distances à l'obstacle dans les trois directions (en face, à droite et à gauche).
- Les sorties sont l'angle de braquage et la vitesse de translation du robot.

Les distances de l'obstacle dans les trois directions permettent de définir 8 situations de base avec les ensembles floues (**P** : proche, **L** : loin).

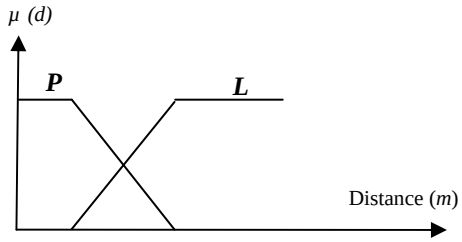


Figure 5. *Les fonctions d'appartenance de la distance*

On a fait deux essais de l'algorithme Q-learning flou avec ces 8 situations. Le premier avec des connaissances imprécises en proposant des interprétations pour les variables linguistique de sortie pour une tache de suivi de mur, et le deuxième sans connaissances à priori.

4.1.1 Suivi de mur avec des connaissances imprécises

Premièrement, la politique adoptée pour cette mission en utilisant le contrôleur flou est exprimée symboliquement par les règles présentées au tableau suivant :

Tableau 1. La matrice d'inférence du suivi de mur

Angle de braquage/ V_r			*La distance d_g*			
			P		**L**	
			La distance d_f			
			P	**L**	**P**	**L**
La distance d_d	**L**	α	*NG*	*NP*	*PG*	*NM*
		V_r	*Z*	*M*	*Z*	*M*
	P	α	*PG*	*Z*	*PG*	*PP*
		V_r	*Z*	*M*	*P*	*P*

Les résultats obtenus sont donnés sur les figures 6-7 :

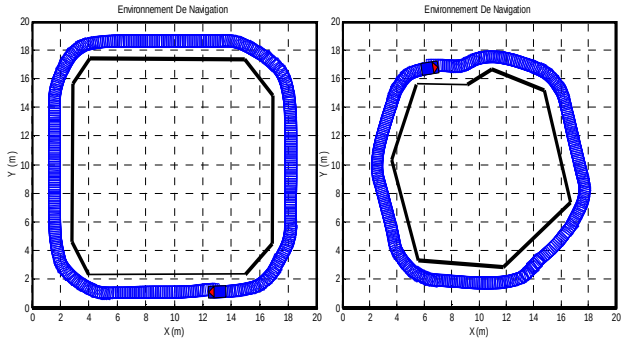


Figure 6 (a-b). *Suivi de mur par le contrôleur flou*

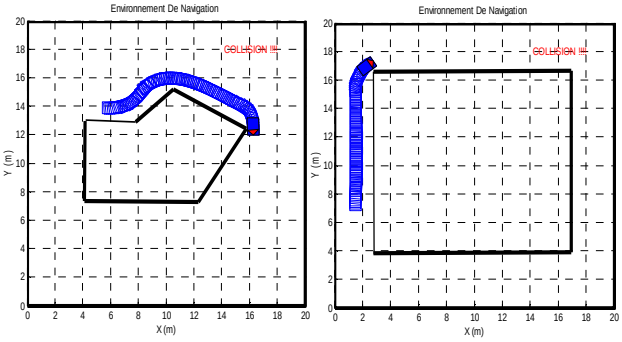


Figure 7 (a-b). *Collision avec les obstacles*

Le contrôleur flou utilisé donne des résultats acceptables pour accomplir cette mission comme montré sur les figures 6 (a-b). Mais dans les cas ou l'obstacle contient des pointes (coins), le comportement est mauvais et le robot ne peut pas éviter les collisions (figure 7 (a-b)).

La base des règles est alors améliorée en ligne en utilisant un signal de renforcement *r* définit par :

$$r = \begin{cases} -2, & \text{Si le robot percute un obstacle,} \\ -1, & \text{Si } d_i < d_m,\ i = 1...3, \\ 0, & \text{ailleurs.} \end{cases}$$

Ce signal va servir à déterminer la meilleure interprétation numérique des termes linguistiques utilisés,

en proposant trois interprétations pour chaque label de sortie (angle de braquage).

La fonction $q[i,j]_{j=1}^{J}$) est associée à l'interprétation j de la règle i qui devient :

Si s est S_i Alors $\alpha = \alpha_{i1}$ avec la qualité $q[i,1]$
ou $\alpha = \alpha_{i2}$ avec la qualité $q[i,2]$
ou $\alpha = \alpha_{i3}$ avec la qualité $q[i,3]$.

Avec : $i = 1...8$ et $j = 1,2,3$

Les figures 8 et 9 montrent des exemples de ce résultat d'optimisation. On observe que le robot est capable d'évoluer dans son environnement sans collision avec les obstacles.

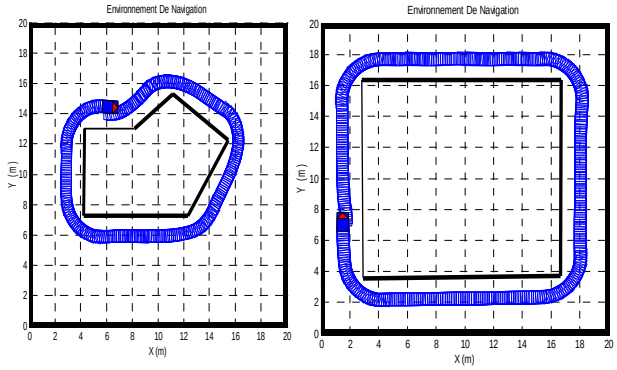


Figure 8. *Les comportements précédents améliorés*

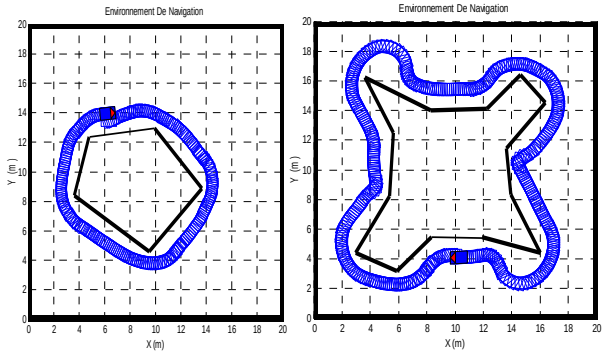


Figure 9. *Suivi des murs de différentes formes*

La moyenne des récompenses obtenues indiquant l'amélioration du comportement du robot est représentée sur la figure 10.

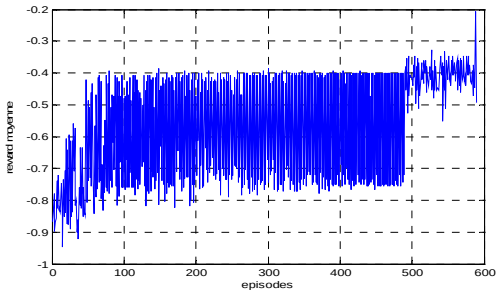


Figure 10. *La moyenne des récompenses obtenues*

4.1.2 Navigation vers la cible avec aucune connaissances à priori

La même base des règles initiales est utilisée, avec le signal de renforcement *r* suivant :

$$r = \begin{cases} -4, \text{ Si le robot percute un obstacle,} \\ -1, \text{ Si } d_i \text{ diminue et } d_i < \frac{l_c}{2}, \\ 0, \text{ ailleurs,} \end{cases}$$

Dans cet essai, la valeur de renforcement va servir à déterminer la meilleure conclusion parmi les trois conclusions proposées pour chaque règle : $\alpha_1 = -\pi/5$, $\alpha_2 = 0$ et $\alpha_3 = \pi/5$.

Dans cette variante, on adopte des règles particulières telles que :

- Au début, le robot est dans une zone libre d'obstacle.
- Dans l'espace libre le robot s'oriente vers la cible.

Les figures 11 et 12 représentent les trajectoires du robot mobile dans les premiers épisodes. On observe des collisions aux obstacles sont produites au début de l'apprentissage.

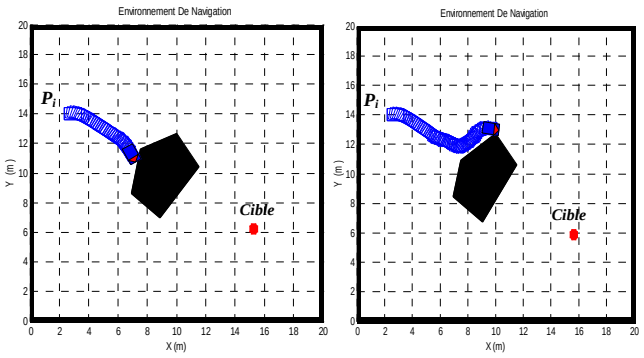


Figure 11. *Episode 1* **Figure 12.** *Episode 20*

Après une phase d'apprentisage, le robot obtient le meilleur comportement pour atteindre la cible. Le robot peut éviter les obstacles et se dirige dans la direction de la cible quelque soit sa position initiale. S'il y a un obstacle devant lui, il choisit l'action tourner à droite.

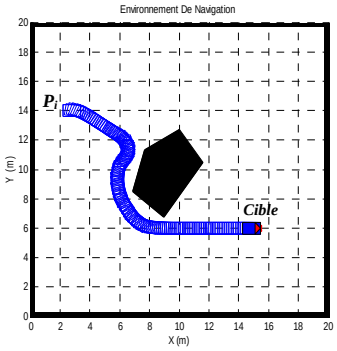


Figure 13. *Trajectoire du robot après la phase d'apprentissage*

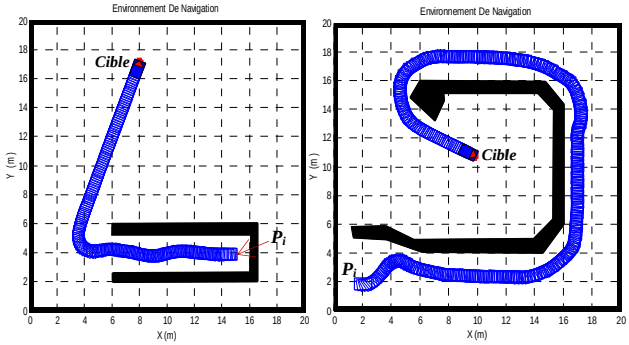


Figure 14 (a-b). *Navigation dans un couloir, suivi de murs*

Les récompenses moyennes reçues sont représentées sur la figure suivante :

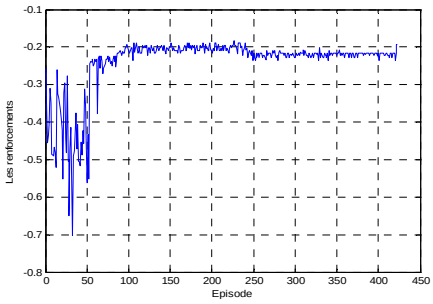


Figure 15. *La moyenne des récompenses obtenues*

5. Conclusion

Dans ce travail, on a présenté une technique intelligente pour l'évitement d'obstacles. C'est une stratégie de commande avec une capacité d'apprentissage en ligne, basée sur la modification automatique du comportement du robot dans son environnement pour maximiser ses récompenses. La méthode nécessite seulement un signal scalaire comme information de retour indiquant la qualité de l'action appliquée. L'algorithme Q-learning flou combine les avantages des deux techniques et considéré comme une extension naturelle de l'algorithme Q-learning de base. L'utilisation des traces d'éligibilités est une amélioration possible de ces deux algorithmes.

6. Références

[1] Patrick Reignier, '' *Pilotage réactif d'un robot mobile, Etude de lien entre la perception et l'action*''. Thèse de Doctorat, institut national polytechnique de Grenoble (INPG), 1994.

[2] Lavi M.Zamstein, A.Antonino Arroyo, Eric M.Schwartz, Sara Keen, Blake C. Sutton, Gorang Gandhi , '' Koolio : Path planning using Reinforcement Learning on a real robot platform''*,.Florida Conference on Recent Advances in Robotics (FCRAR).* Miami, 2006.

[3] William D. Smart, Leslie Pack Kaelbling "Effective Reinforcement Learning for Mobile Robots". *Proccedings of 2002 IEEE. International Conference on Robotics & Automations, Washington, DC,* 2002.

[4] Richard S. Sutton and Andrew G. Barto,Ronnald J.Williams, '' Reinforcement Learning is direct adaptive Optimal Control '', *Proc of ACC,* Boston, June 1991.

[5] Richard S. Sutton and Andrew G. Barto, '' *Reinforcement Learning: An Introduction* ''. MIT Press, Cambridge, Massachusetts, 2005.

[6] David Filliat, '' *Robotique Mobile* '', C10-2 Ecole Nationale Supérieur des Techniques Avancées ENSTA, Octobre 2004.

[7] Stephan ten Hagen and Ben Krose , '' Q-Learning for systems with continuous state and action spaces''. *Proc BENELEARN 10th 2000, Belgian-Dutch Conference in Machine Learning*, 2000.

[8] Claude Touzet, '' L'apprentissage par Renforcement'', *CESAR, USA*, Janvier 1998.

[9] Glorennec Pierre Yves, '' *Algorithmes d'Apprentissage Pour Systèmes d'inférence Floue*''. Editions Hermes, 1999.

[10] Boumehraz Mohamed & al, '' Fuzzy Inference Systems Optimization by Reinforcement Learning''. *Courrier du Savoir – N°01*, pp. 09-15, Université de Biskra, Algérie, Novembre 2001.

[11] Nelson H.C.Yung and Cang Ye, '' An Intelligent Mobile Vehicle Navigation Based on Fuzzy Logic and Reinforcement Learning''. *IEEE Transactions on Systems, Man and Cybernetics*, vol 29. no.2, pp.314-321, 1999.

[12] Cang Ye, Nelson, H.C,Yung, Danwei Wang, '' A Fuzzy Controller with Supervised Leaning Assisted Reinforcement Learning Algorithm For Obstacle Avoidance''. *IEEE Transaction on Systems, Man, And Cybernetics-Part B: Cybernetics*, vol.33, no.1,pp.1-11, avril 2003.

[13] Parthasarathi Rishikesh1, Lavanya Janardhanan and Er Meng Joo, '' Goal seeking of Mobile robots using dynamic fuzzy Q-learning ''**.** *Journal of the Institution of Engineers*, Singapore vol. 45 Issue 5, 2005.

[14] Kaelbling,L.P.,Littman,M.L., and Moore,A.W, '' Reinforcement Learning: A survey'', *Journal of Artificial Intelligence Research*, vol 4,pp.237-285, 1996.

[15] Glorennec Pierre Yves,'' Reinforcement Learning: an Overview ''**.** *ESIT 2000, Aachen, Germany*, 14-15 septembre 2000.

[16] Watkins C., Dayan P. '' Technical Note, Q-Learning ''**,** *Machine Learning*, 8, p.279-292, 1992.

[17] Kevin M. Passino et Stephen Yurkovich, '' *Fuzzy Control* '', Addison Wesley Longman, Inc. Department of Electrical Engineering, The Ohio State University, 1998.

[18] M.Sc. Mykhaylo Konyev , '' Using fuzzy Inference System as a function approximator of a state action table''. *Advanced Aspects of Theoretical Electrical Engineering*, Sozopol, Bulgaria, 2005.

[19] Glorennec Pierre Yves, Lionnel Jauffe, '' A Reinforcement Learning Method for an Autonomious Robot''. *Proceedings of the First Online Workshop on Soft Computing*, 1996.