Conférence Internationale en Génie Electrique CIGET 09, 25-26 Octobre 2009, Tébessa, Algérie

Navigation d'un robot mobile en utilisant le Q-learning flou

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

Résumé

Ce travail présente une technique de navigation d'un robot mobile. C'est une méthode basée sur l'apprentissage à partir de l'expérience de manière à maximiser une récompense numérique au cours du temps. L'idée fondamentale de l'apprentissage par renforcement est d'améliorer une politique courante après chaque interaction avec l'environnement. Tout d'abord, l'algorithme Q-learning standard de l'apprentissage par renforcement est utilisé pour la navigation d'un robot mobile. Pour améliorer les performances, une stratégie de commande avec une capacité d'apprentissage en ligne est utilisée en introduisant des connaissances disponibles à priori par un système d'inférence flou pour que le comportement initial soit acceptable. C'est une extension de Q-learning aux espaces d'états et d'actions continus pour la navigation d'un robot mobile.

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

1. Introduction

La navigation d'un robot mobile peut être considérée comme la tache de déterminer une trajectoire permettant au robot de se déplacer d'une position initiale vers une autre finale désirée en évitant les obstacles, tout en respectant les contraintes cinématiques du robot et sans intervention humaine. Le problème de déterminer une telle trajectoire est connu aussi sous le nom du problème de planification de trajectoire [1].

Contrairement aux méthodes de navigation traditionnelles qui nécessitent une planification complète de l'environnement, les stratégies de commande réactive offrent des solutions intéressantes qui utilisent directement l'information issue du capteur du robot pour accomplir ses objectifs [1][2].

Initialement, la destination finale du robot est exprimée dans un environnement 2D par les coordonnées (x_c, y_c). Malheureusement, ce n'est pas toujours le cas dans les applications en extérieur comme le transport, l'agriculture, la surveillance, etc, ou la position finale est souvent requise avec une bonne précision. De ce fait, ces taches se traduisent alors par une navigation autonome de la position initiale (x_i, y_i, θ_i) vers un but orienté (x_c, y_c, θ_c) .

La programmation d'une stratégie de navigation autonome passe par l'emploi d'une commande intelligente, nécessitant des connaissances précises d'un expert. L'acquisition des pairs état-action est une tache importante dans la commande des robots mobiles [3]. L'application de l'apprentissage par renforcement est une solution possible pour le problème de navigation d'un robot mobile puisqu'il ne nécessite pas des exemples d'apprentissage. Le signal 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 [4][5][6].

Malheureusement l'apprentissage par renforcement est parfaitement adapté aux processus dont le nombre d'états et d'actions est limité [7]. Dans le cas de processus continus dont le nombre d'état et d'action est infini, il est nécessaire d'estimer correctement la fonction de qualité en utilisant les approximateurs universels comme les systèmes d'inférence floue et les réseaux de neurones [3][8][9].

Ce papier est organisé comme suit : dans la section 2, un bref exposé sur l'apprentissage par renforcement et ses méthodes sera présenté. La section 3 présente l'algorithme Q-learning et son application pour une tache de recherche de cible. Une version floue de cet algorithme qui appelée le Q-learning flou est utilisée ensuite pour améliorer les performances du robot et d'utiliser des espaces d'états et d'actions continus pour une de tache de navigation d'un robot mobile (section 4). Dans la section 5, la conclusion récapitule ce qui a été fait et présente les perspectives envisagées de ce travail.

2. L'apprentissage par renforcement

L'apprentissage par renforcement est une approche de l'intelligence artificielle qui permet l'apprentissage d'un agent par l'interaction avec son environnement; 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 [7][10]. C'est une méthode de programmation ne nécessitant que de spécifier les moments auxquels punir ou récompenser l'agent. Il n'est nul besoin de lui indiquer que faire dans telle ou telle situation, l'agent se charge de l'apprendre par lui même en renforçant les actions qui s'avèrent les meilleures [11].

Si on considère un agent situé dans un environnement et avec lequel il peut interagir. L'agent à l'instant t perçoit d'une part cet environnement (*situation* s_t), et peut agir (*action* a_t), et d'autre part ; reçoit une récompense (r_t), comme illustré sur la figure.1, ou s_{t+1} *et* r_{t+1} sont la situation et la récompense suivante.

L'agent ne connaît pas la situation dans laquelle il se trouve que par l'intermédiaire de l'historique de ses perceptions. Son but, dans le cadre de l'apprentissage par renforcement, est de trouver le comportement le plus efficace, c'est à dire savoir, dans chaque situation possible, quelle action accomplir pour maximiser l'espérance de ses gains [7][12] :

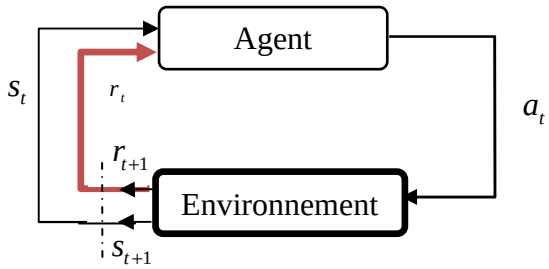


Figure.1 *Système d'apprentissage par renforcement*

Le comportement de l'agent est défini par une *politique* $\pi : \{S, A\} \to [0,1]$, qui guide l'agent de manière probabiliste en spécifiant, pour chaque état s la probabilité de réaliser l'action a (et donc $\pi(s) = a$). Le but de l'agent est de trouver la politique optimale π^* maximisant la récompense à long terme (à priori stochastique) [7].

2.1 Fonction de valeurs

Pour mesurer la performance du comportement d'un agent qui est associé naturellement à un problème de la forme d'un processus de décision markovien PDM ; on utilise l'appellation de la fonction de valeur qui est l'espérance de gain si l'on suit la politique π à partir de l'état courant s [7].

La fonction de retour est :

$$R_t = \sum_{k=1}^{\infty} \gamma^k r_{t+k} \qquad (1)$$

et la fonction de valeur est alors :

$$V_\pi(s) = E_\pi(R_t | s_t = s) = E_\pi\left(\sum_{k=1}^{\infty} \gamma^k r_{t+k} | s_t = s \right) \qquad (2)$$

Ou $\gamma \in]0,1[$ est un facteur de dépréciation.

2.2 Fonction de qualité

En plus d'état, d'action, fonction de récompense, et de fonction de probabilité de transition, un autre concept important de PDM est la fonction de qualité, qui inclut la fonction de valeur d'état et d'action [7][13]. Elle est définie pour une politique fixée π par:

$$Q^\pi(s,a) = E_\pi(R_t | s_t = s, a_t = a) \qquad (3)$$

La fonction de qualité d'une politique permet de donner pour chaque état, le retour futur si l'agent suit cette politique, et permet également de comparer les politiques en définissant un ordre partiel [7][14]:

$$\pi \geq \pi' \Leftrightarrow \forall s, V^\pi(s) \geq V^{\pi'}(s) \qquad (4)$$

Cet ordre permet de définir la fonction de valeur de la politique optimale que l'apprentissage par renforcement va cherche à estimer :

$$V^*(s) = \max_\pi V^\pi(s) \qquad (5)$$

C'est une fonction qui peut aussi s'exprimer pour une paire état-action :

$$Q^*(s,a) = \max_\pi Q^\pi(s,a) \qquad (6)$$

Avec la relation suivante :

$$Q^*(s,a) = E(r_{t+1} + \gamma V^*(s_{t+1}) | s_t = s, a_t = a) \qquad (7)$$

Les méthodes de l'apprentissage par renforcement peuvent être groupées en deux catégories : les méthodes basées modèle et à modèles libres. La première catégorie a une liaison directe avec la programmation dynamique, et la deuxième n'exige pas la disponibilité d'un modèle de l'environnement [7].

2.3 La programmation dynamique

Elle permet de calculer une politique optimale dans un PDM connu, en utilisant les propriétés des équations de Bellman, ou le modèle de l'environnement est connu. Ces algorithmes permettent d'estimer la fonction de valeur optimale V^* afin d'en déduire une politique optimale [7].

On trouve deux méthodes qui sont l'itération de valeur et l'itération de politique qui utilisent l'équation suivante comme étape de mise à jour pour calculer une suite de fonctions V_k^π qui convergera vers V^π [11]:

$$V_{k+1}^\pi(s) = \sum_{a \in A(s)} \pi(s,a) \sum_{s' \in S} P(s,a,s') \left[R(s,a,s') + \gamma V_k^\pi(s') \right] \qquad (8)$$

2.4 Les méthodes à modèles libres

Ce sont les méthodes les plus utilisées en pratique et ne nécessitent pas un modèle de l'environnement. Il existe de nombreux algorithmes basés sur la différence temporelle comme TD(0), SARSA, l'acteur critique (ACRL) et le Q-learning. Ce dernier est l'algorithme le plus connu et le plus utilisé. Toutes ces méthodes sont basées sur les processus de décision markoviens et demandent de discrétiser l'environnement en états et les consignes sur les actionneurs en commandes échantillonnées [7].

3. Le Q-learning

Il s'agit sans doute de l'algorithme d'apprentissage par renforcement le plus connu et le plus utilisé en raison des preuves formelles de convergence [13]14].

Le principe du Q-learning consiste à apprendre la fonction de qualité en interagissant avec l'environnement 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}) [7][11]. Cette mise à jour se fait sur la base de l'observation des transitions instantanées et de leur récompense associée par l'équation suivante :

$$Q(s_t,a_t) \leftarrow Q^{\pi}(s_t,a_t) + \alpha\left[r_t + \gamma \max_{a \in A(s)} Q(s_{t+1},a_t) - Q(s_t,a_t)\right] \quad (9)$$

Où $\alpha \in [0,1]$ est un cœfficient d'apprentissage qui doit être initialement proche de 1 et diminue ensuite pour tendre vers 0.

L'algorithme peut être résumé comme suit :

$Q(s,a) \leftarrow 0, \ \forall (s,a) \in (S,A)$

Pour un grand nombre ∞ faire

Initialiser l'état initial s_0

$t \leftarrow 0$

Répéter

Choisir l'action à émettre a_t en fonction de la politique de Q et l'émettre. Observer r_t et s_{t+1}.

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

$t \leftarrow t+1$

Jusqu'à $s_t \in F$

Fin pour

L'algorithme *Q-learning*

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

On a utilisé l'algorithme précédent pour une tache de recherche d'une cible par un robot mobile. Le robot construit une fonction Q qui définit la qualité de chaque action appliquée dans un état donné. Dans le cas discret c'est un tableau, qui comporte des coefficients qui représentent la qualité de telle ou telle action. Après l'exécution d'une action, les coefficients doivent être ajustés en fonction du résultat obtenu. Cela se fait par le biais de la formule (9).

L'espace autour du robot est divisé en secteurs selon l'angle entre l'orientation du robot et celle de la cible notée *E_ang*, et la distance entre le robot et la cible notée *E_pos* ou l'erreur de position. Les actions utilisées sont : avancer, tourner à droite et tourner à gauche. Ces actions sont choisies par la politique d'exploration-exploitation (*PEE)*. Ce qui permet au robot d'explorer l'espace d'états.

Pendant le déplacement, le robot reçoit les valeurs suivantes comme signaux de renforcement:

$$r = \begin{cases} 4, & \text{Si le robot atteint la cible.} \\ 3, & \text{Si } E\_pos \text{ déminue et } E\_ang = 0. \\ 2, & \text{Si } E\_pos \text{ et } E\_ang \text{ déminuent.} \\ -1, & \text{Si } E\_pos \text{ déminue et } E\_ang \text{ augmente.} \\ -2, & \text{Si } E\_pos \text{ augmente.} \\ -3, & \text{Si le robot percute les murs de son environnement.} \end{cases}$$

Afin de généraliser la navigation du robot mobile pour toute les situations, l'apprentissage se fait avec une position initiale aléatoire du robot et de la cible dans chaque épisode.

Les figures suivantes 2 (a-b-c-d) présentent des trajectoires du robot obtenues après une phase d'apprentissage. On remarque que le robot se dirige vers la cible quelque soit sa posotion initiale.

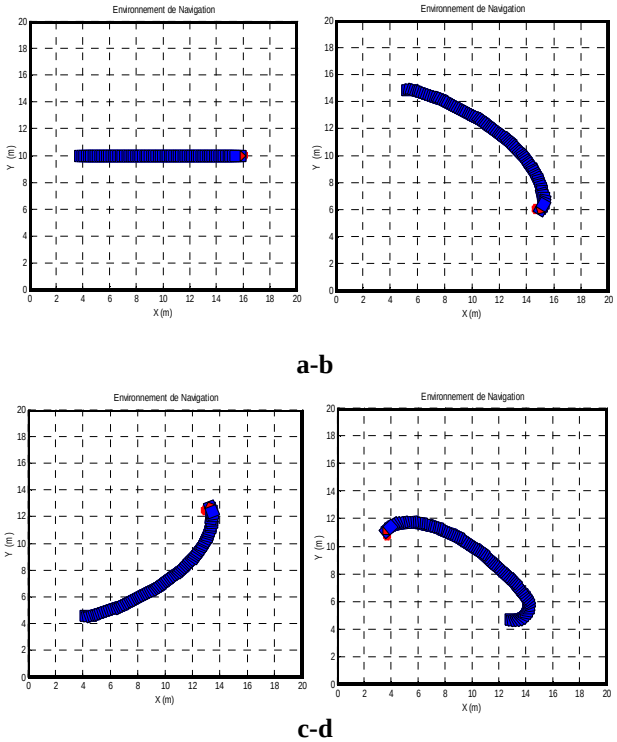


Figure.2 *Navigation libre en utilisant le Q-learning*

Comme indicateur d'apprentissage, la figure 3 présente la moyenne des récompenses obtenues par le robot dans chaque épisode.

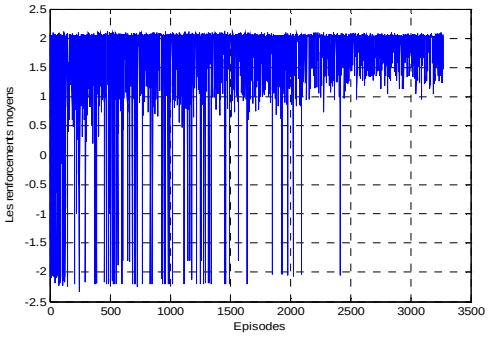


Figure.3 *Les valeurs moyennes du renforcement*

Plusieurs implémentations de l'algorithme Q-learning ont été réalisées en variant le nombre d'états et d'actions proposées. Le tableau suivant décrit le nombre des épisodes en fonction des espaces utilisés pour obtenir un comportement acceptable.

Nombre d'états	Nombre d'actions	Nombre d'épisode
09	03	3260
11	03	4100
13	03	6200
19	05	8950

Tableau l

Une discrétisation plus fine permet d'améliorer le comportement du robot, mais nécessite un volume mémoire plus important et un temps d'apprentissage plus long.

Dans une tache de navigation d'un robot mobile, l'espace d'états et d'actions sont continus. Dans ce cas, l'utilisation d'autres approches pour approximer les fonctions de qualités est une solution possible. Dans la section suivante, on va utiliser les systèmes d'inférences floue pour introduire des connaissances à priori pour que le comportement initial soit acceptable et afin d'accélérer l'apprentissage.

4. Le Q-learning flou

Il existe plusieurs extensions du Q-learning pour les espaces d'états et d'actions continus en utilisant les systèmes d'inférence floues [3][9][15]. La tâche est d'approcher la fonction de qualité Q par la fonction :

$$s \rightarrow y = \hat{Q} = SIF\ (s) \qquad (10)$$

Le principe de cette extension consiste à 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 [3][9][11].

La base des règles initiale en utilisant un modèle flou de type Takagi-Sugeno d'ordre 0 est composée de m règles de la forme [3][15][17]:

$$\begin{aligned} \text{Si } s \text{ est } S_i \text{ Alors } & y = a[i,1] \text{ avec } q[i,1] = 0 \\ \text{ou } & y = a[i,2] \text{ avec } q[i,2] = 0 \\ & \ldots \\ \text{ou } & y = a[i,N] \text{ avec } q[i,N] = 0 \cdot \end{aligned} \qquad (11)$$

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 donnée $PEE(i) \in \{1...N\}$. Dans ce cas, la sortie inférée est donnée par :

$$A(s) = \sum_{i=1}^{m} w_i(s).a[i, PEE(i)] \qquad (12)$$

La qualité de cette action sera :

$$\hat{Q}(s, A(s)) = \sum_{i=1}^{N} w_i(s).q[i, PEE(i)] \qquad (13)$$

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

Dans cet algorithme, le robot possède une base de règles initiale qui définit les situations possible pour la tache de recherche d'une cible. Le contrôleur utilise l'angle entre l'orientation du robot et la cible notée *E_ang* et la distance entre la position du robot et celle de la cible notée *E_pos* pour générer l'angle de braquage α.

Les fonctions d'appartenance pour *E_pos* et *E_ang* sont :

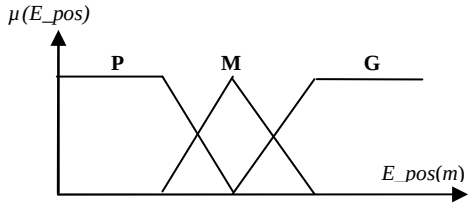


Figure.4 *Les fonctions d'appartenance pour E_pos*

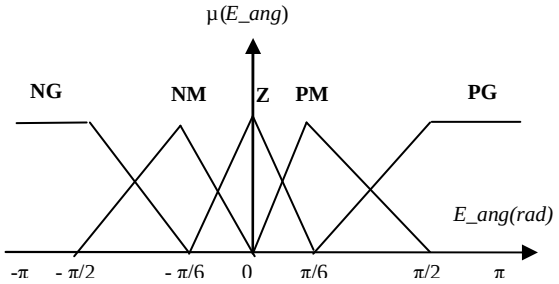


Figure.5 *Les fonctions d'appartenance pour E_ang*

Avec les variables linguistiques suivantes :

P : Proche **M :** Moyen **G :** Grand

Z : Zéros **PM** : Positive Moyenne **PG** : Positive Grande

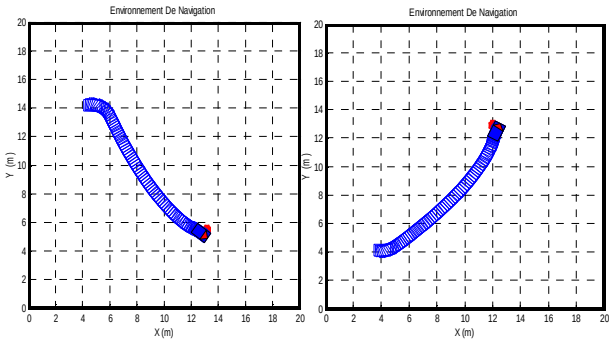
NG : Négative grande **NM** : Négative Moyenne

Pendant la navigation, le robot reçoit les mêmes valeurs de renforcement utilisé dans la section précédente.

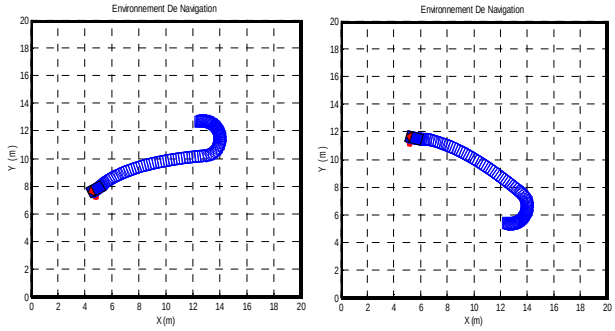
Durant l'apprentissage et pour optimiser le système de navigation utilisé, les positions initiales sont choisies aléatoirement ou chaque épisode commence avec une position aléatoire et se termine lorsque le robot atteint la cible ou percute les limites de son environnement.

Pour chaque règle, 3 conclusions sont proposées. Après une phase d'apprentissage, le robot choisit pour chaque règle la conclusion correspondante à la meilleure qualité $q[i, j]_{j=1}^{N}$.

Les trajectoires du robot mobile après une phase d'apprentissage sont représentées sur les figures 6 (a-b-c-d). La valeur moyenne des renforcements reçus pour chaque épisode est présentée sur la figure 7.



a-b



c-d

Figure.6 *Navigation libre en utilisant le Q-learning flou*

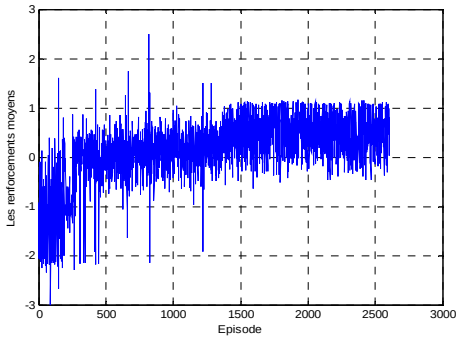


Figure.7 *Les valeurs moyennes de renforcement*

4.2 Utilisation des traces d'éligibilités

Pour tenir compte de l'effet des actions précédentes sur le comportement, on utilise une valeur auxiliaire appelée *trace d'éligibilité* que l'on définit de la manière récursive suivante [7][16].

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & si \quad s \neq s_t \\ \gamma\lambda e_{t-1}(s) + 1 & si \quad s = s_t \end{cases} \qquad (14)$$

L'utilisation de cette notion est considérée comme une généralisation des algorithmes. Cette éligibilité est quantifiée par le niveau de responsabilité : le dernier état ou la dernière paire état-action est la plus responsable, donc possède l'éligibilité la plus grande.

Les trajectoires du robot en utilisant les traces d'éligibilités avec 15 règles et 3 conclusions proposées sont représentées sur les figures 8 (a-b-c-d). On observe l'amélioration du comportement du robot comme le montre la figure des valeurs moyennes des renforcements reçus. Dans tous les cas, le robot se dirige vers la cible quelque soit sa position initiale. L'apprentissage en utilisant les traces d'éligibilités est plus rapide.

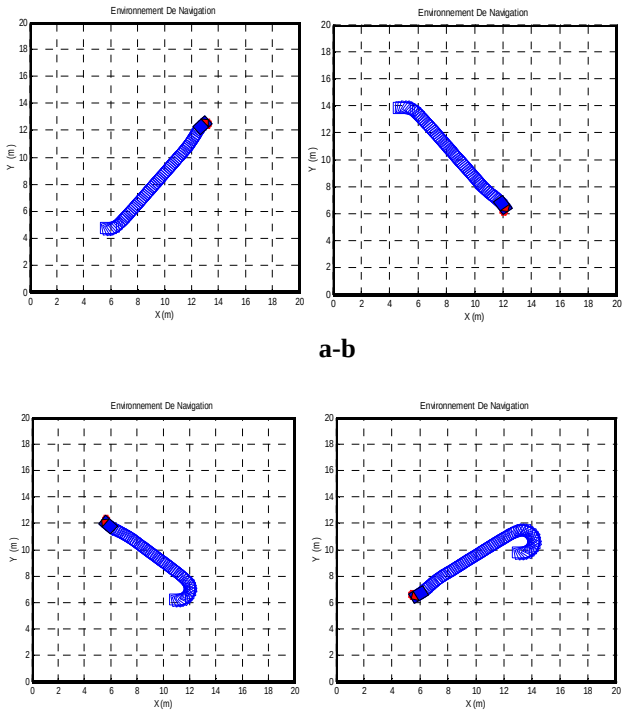


Figure.8 *Navigation libre en utilisant les traces d'éligibilités*

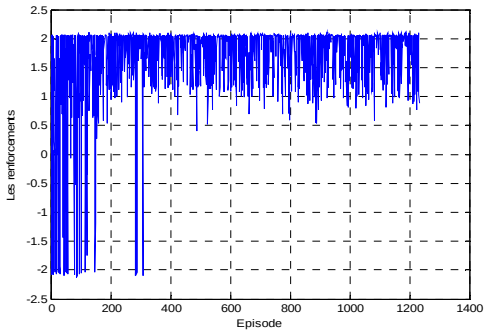


Figure.9 *La moyenne des récompenses*

5. Conclusion

Dans cet article, on a présenté une méthode de navigation intelligente d'un robot mobile. 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. Le principe du Q-Learning flou consiste à optimiser les conclusions des règles floues en utilisant un signal de renforcement. Ce signal permet au navigateur d'ajuster sa stratégie pour améliorer ses performances. Cet algorithme combine les avantages des deux techniques et considéré comme une extension naturelle de l'algorithme Q-learning de base. Pour accomplir les objectifs confiés au robot mobile, le robot doit avoir une capacité d'évitement d'obstacles pour évoluer dans les environnements restreints.

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] Bernard Bayle, '' *Robotique Mobile* '', Ecole Nationale Supérieure de Physique de Strasbourg, Université Louis Pasteur, 2007.

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

[4] 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.

[5] 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.

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

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

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

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

[10] Richard S. Sutton, '' Learning to predict by the Methods of Temporal Differences" .*Machine Learning*,3,p.9-44, 1988.

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

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

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

[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] 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.

[16] Richard S. Sutton, '' Reinforcement Learning with replacing eligibility Traces '', *Machine Learning*, 22,123-158, 1996

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