Sélection de l'action par apprentissage dans une carte de planification: application aux environnements dynamiques

M. Quoy, P. Gaussier, S. Leprêtre, A. Revel[1]

J.P. Banquet[2]

Résumé : Nous présentons ici un modèle neuronal permettant de planifier les trajectoires d'un robot mobile. Ce modèle est fondé sur l'élaboration d'une "carte cognitive" reliant par un graphe les lieux à atteindre. Nous montrons d'abord que cette carte peut être utilisée pour contrôler la vitesse d'approche d'un but. Puis nous montrons que ce modèle permet de choisir entre différents buts contradictoires dans un environnement dynamique.

Abstract: We present an artificial neural model for planning trajectories of a mobile robot. It is based on the building of a "cognitive map" linking together places that may be reached successively. We first show that this map may be used for controlling the speed when approaching a goal. Then we show that this architecture enables to choose between different contradictory goals in a dynamical environment.

Mots-clés : Sélection de l'action, apprentissage, carte cognitive, robot autonome

[1]Université de Cergy-Pontoise-ENSEA, Groupe neurocybernétique-ETIS, 6, av du Ponceau 95014, Cergy-Pontoise, France, e-mail: quoy@ensea.fr

[2]Institut Neurosciences et Modélisation, Paris VI

1. Introduction

Notre axe de recherche principal est la définition d'architectures de contrôle pour des robots mobiles. Le but est de doter ces robots d'une totale autonomie de manière à ce qu'ils naviguent et réalisent des tâches seuls dans un environnement inconnu (*approche animat* [MEY 91, DON 96]). Pour cela, nous prenons inspiration de résultats neuro-psychologiques et neuro-biologiques aussi bien pour l'architecture que pour les mécanismes d'apprentissage. L'architecture adoptée est donc fondée sur des réseaux de neurones artificiels (R.N.). Elle permet en outre d'adopter une *approche constructiviste* en "ancrant" les représentations sur les données sensorielles (essentiellement la vision). Notre architecture est constituée de deux parties. La première permet à l'animat de se situer dans son environnement. Nous ne détaillerons pas cette partie dans cet article. La seconde construit un graphe reliant des lieux successivement atteints. Ce graphe est notre carte de planification. En plus de ce mécanisme, il y a mécanisme réflexe qui prend contrôle de l'animat quand un obstacle est rencontré, et qui le suit. Il est important de souligner qu'aucune information n'est donnée *a priori* sur l'environnement. Le coeur de l'architecture est l'apprentissage de nouveaux lieux et de leurs liens avec des lieux déjà appris. La carte ainsi créée est représentative d'un environnement particulier. Elle permet à l'animat de résoudre des problèmes de sélection de l'action dans un environnement statique [GAU 98]. Cependant, la carte peut être modifiée de manière à fonctionner encore si l'environnement a changé (porte fermée, personne qui bouge ...). Les autres changements possibles sont apportés par les mouvements de l'animat. Ils peuvent être aléatoires (exploration de l'environnement) ou bien dictés par la carte de planification (quand l'animat doit atteindre un but). En utilisant un formalisme de champ neuronal (*approche dynamique* [SCH 95]), la carte de planification donne la consigne de vitesse en vue d'atteindre un but. Par la suite, nous allons d'abord décrire l'environnement expérimental et l'architecture de planification. Ensuite nous montrerons comment la carte de planification peut être utilisée dans une approche dynamique pour contrôler la vitesse. Finalement, nous présenterons des résultats sur la manière dont l'animat résout des problèmes de sélection de l'action en trouvant des chemins dans des environnements dynamiques.

2. La simulation réalisée

Les expériences décrites dans cet article ont été réalisées dans un environnement simulé. Certaines d'entre elles ont déjà été réalisée sur un robot réel. L'animat est seulement guidé par la vision. Il ne voit que des points de repère (amers) et des obstacles. Il ne voit pas les endroits où il peut trouver des

ressources. L'environnement a une largeur d'au maximum 40 tailles d'animat. Deux niveaux d'énergie différents décroissent exponentiellement avec le temps, sauf s'ils sont réapprovisionnés par une source appropriée [GAU 98]. Comme nous sommes dans une approche animat, nous appellerons par la suite ces niveaux "nourriture" et "eau". Ils sont liés aux motivations "faim" et "soif". Dans un cadre plus proprement robotique, on parlerait plutôt de "station de recharge", "lieu de décharge" ... Une exploration aléatoire de l'environnement permet à l'animat de découvrir des lieux d'approvisionnement. Il doit aussi aller à une source d'approvisionnement appropriée quand il en ressent le besoin. Quand une source est vide, une autre réapparaît aléatoirement dans l'environnement; l'animat doit donc toujours explorer pour trouver de nouvelles sources. L'animat doit ainsi résoudre le dilemme entre explorer (pour trouver de nouvelles sources d'approvisionnement), aller vers une source connue, et se reposer. Nous imposons en outre le fait que les niveaux des différentes variables doivent être maintenus entre deux seuils qui définissent la "zone de confort", de telle sorte que dès qu'un niveau est en dessous de ce seuil l'animat se dirige vers la source appropriée. A l'inverse, si le niveau se trouve au dessus du deuxième seuil, l'animat s'éloigne de la source de manière à lui faire explorer des zones plus lointaines. L'animat réussit parfaitement à maintenir les niveaux dans la zone de confort. L'animat n'échoue que lorsqu'une source apparaît dans une zone difficile à explorer.

3. Modèle

Nous ne détaillerons pas ici notre modèle de navigation (voir [GAU 95, GAU 97] pour une présentation plus complète). Il faut juste retenir qu'un lieu est défini par un ensemble de paires (amer, azimut). Quand l'animat explore l'environnement, il apprend de nouveaux lieux qui sont suffisamment différents de ceux précédemment appris. Un neurone code pour chaque lieu appris dans l'environnement. Plus l'animat est proche de ce lieu, plus la réponse de ce neurone est importante. Nous rappelons que nous ne donnons aucune description de l'environnement, ni que nous n'apprenons quoi faire pour chaque endroit de l'environnement. S'il veut retourner à un lieu appris, l'animat doit conserver la trace de l'endroit où il était et des lieux qu'il peut rejoindre depuis chaque endroit appris. Il faut donc introduire une sorte de carte de l'environnement pour résoudre ce problème. Nous ajoutons donc aux neurones précédents un groupe de neurones capables d'apprendre les relations entre des lieux visités successivement. La proximité temporelle étant équivalente à une proximité spatiale, le système crée une représentation topologique de son environnement. Nous appelons ce groupe carte de planification (ou "carte cognitive") (fig. 1 et 2) [TOL 48, THI 96, REV 98].

Au niveau planification (fig. 1), les motivations activent directement les neurones de planification liés au but à atteindre (l'intensité du neurone de planification est proportionnelle à la motivation). Cette activité est ensuite diffusée de neurone à neurone par l'intermédiaire des poids synaptiques. Sur la figure 1 est présenté un exemple de diffusion de la motivation "A". Cette diffusion permet de trouver le chemin le plus court dans ce graphe. Il est prouvé que cet algorithme trouve tout le temps le plus court chemin dans le graphe (il est équivalent à l'algorithme d'exploration de graphe proposé par Bellman-Ford [BEL 58]).

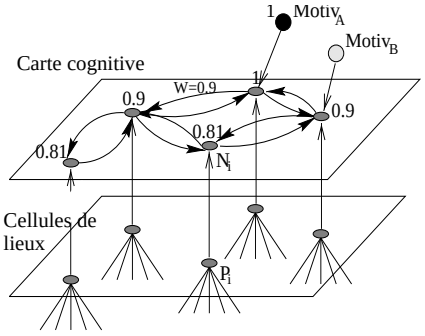

Figure 1. *Architecture globale du système de planification. Les lieux sont reconnus au niveau "Cellules de lieux". Les neurones du niveau but codant des lieux qui peuvent être rejoints sont liés entre eux. Quand une motivation "A" active un but, l'information est propagée sur le graphe de neurone en neurone dans toutes les directions (ici tous les poids valent* 0.9*).*

L'animat essaie de suivre le gradient de l'activité des neurones de la carte de planification pour choisir le lieu suivant à atteindre. Le but ou sous-but le plus intense dans le voisinage de l'animat est alors sélectionné et attire l'animat. Quand l'animat est suffisamment proche de cet endroit, le sous-but associé est inhibé, et l'animat est attiré par le sous-but suivant, et ainsi de suite jusqu'à ce que le but soit atteint. Le principe de ce genre de cartes cognitives n'est pas nouveau en soi [ARB 77, SCH 92b, BAC 94, BUG 95, TRU 97, FRA 98]. La nouveauté dans ce article est que notre algorithme permet de résoudre des problèmes de planification impliquant plusieurs buts évoluant dans le temps dans un environnement qui peut lui-même changer (les sources peuvent disparaître et apparaître autre part, des obstacles peuvent bouger, des amers peuvent être masqués ...). La carte est apprise et modifiée en temps réel et permet de satisfaire des cas où les buts sont contradictoires. Les sous-buts correspondent aux deux cas suivants : la fin d'un évitement d'obstacles (par exemple l'animat mémorise le chemin entre deux pièces (position de la porte)), ou bien des lieux mal reconnus (l'activité des neurones est en dessous d'un certain seuil). Plus ce seuil est élevé, plus il y a de sous-buts appris. On obtient alors un pavage quasi régulier de l'environnement (fig. 2).

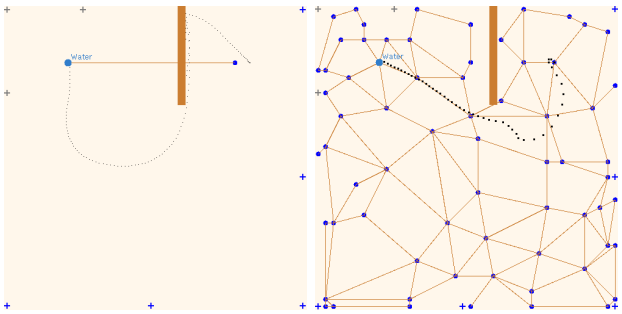

Figure 2. *A gauche, mécanisme réflexe de suivi d'obstacles afin d'atteindre le but (la vitesse initiale est vers le haut à gauche. L'animat possède une inertie). Les pointillés indiquent la trajectoire suivie par l'animat. Sur la droite, carte cognitive construite par exploration du même environnement. Les amers sont les croix sur les bords. Chaque cercle est un sous-but. Les liens indiquent les sous-buts qui peuvent être atteint successivement.*

L'apprentissage de la carte cognitive a lieu tout le temps. Il n'y a pas de séparation entre les phases d'apprentissage et d'utilisation. Les liens entre neurones successifs dans le graphe peuvent être renforcés (apprentissage hebbien). La règle d'apprentissage est la suivante :

$$\frac{dW_{i,j}}{dt} = -\lambda.W_{i,j} + (C + \frac{dR}{dt}).(1 - W_{i,j}).\overline{G_i}.G_j \qquad [1]$$

W_{ij} est le poids associé à la possibilité de relier le lieu j depuis le lieu i ($0 < W_{ij} < 1$). G_i est l'activité d'un neurone. Elle est non nulle jusqu'à ce que G_j (avec $i \neq j$) soit activé par la reconnaissance du lieu j. Ceci est réalisé par une intégration temporelle de G_i représentée dans l'équation par $\overline{G_i}$. $\overline{G_i}$ décroît avec le temps et peut être considérée comme une mesure sommaire de la distance entre i and j. λ est une constante positive très petite. Elle permet l'oubli de liens inutilisés. Dans nos simulations $C = 1$. Le terme $\frac{dR}{dt}$ correspond à la variation du signal de renforcement externe (négatif ou positif) qui apparaît quand l'animat entre dans un lieu "dangereux" ou en sort [GAU 98].

4. Utilisation de la carte de planification pour le contrôle de la vitesse

La carte de planification indique quel chemin suivre pour rejoindre un but. Elle peut aussi donner une autre information concernant le contrôle de la vitesse afin d'assurer une convergence vers le but et un arrêt juste dessus. La carte de planification indique donc non seulement où est le but, mais aussi comment le rejoindre. A cet effet, nous allons considérer que l'animat a une masse M et que

le déplacement s'effectue avec un frottement F [SCH 92a]. Afin de compenser le frottement, une force interne est présente. Lorsque l'animat explore, l'intensité et la direction de cette force sont aléatoires. Lorsque l'animat veut rejoindre son but, la direction de la force est la direction du sous-but à rejoindre. Son intensité peut être constante (indépendante de la distance au but), ou bien dépendre de la distance au but (gradient de G). Les équations contrôlant le mouvement de l'animat deviennent alors :

$$\begin{array}{l} v_x(t+1) = v_x(t) - \frac{dt.F.v_x(t)^2}{M} + \frac{dt.drive}{M} \\ x(t+1) = x(t) + dt' v_x(t) \end{array} \qquad [2]$$

où l'indice x indique une projection sur l'axe des x (même équation sur y). v est la vitesse et (x, y) la position de l'animat. $F = 5$, $M = 5$ et $drive$ une force (aléatoire ou vers le but). Les valeurs de ces constantes ont été choisies pour montrer un comportement de glissement permettant d'illustrer clairement l'influence de $drive$ sur le mouvement de l'animat. dt et dt' sont des constantes d'intégration. Pour rester au même endroit (sur le but par exemple), la vitesse doit être nulle. Ainsi à cet endroit, $drive = 0$. Par ailleurs, si $drive$ est constant, la vitesse tend vers $\sqrt{\frac{drive}{F}}$ et le robot ne peut pas s'arrêter sur le but. Cependant, si $drive$ dépend de la distance au but, on peut faire tendre $drive$ vers 0 quand l'animat se rapproche du but. Dans le cas où $drive$ n'est pas constant, la différence entre la sous-but le plus proche et le sous-but suivant donne l'intensité de cette force (gradient). Cette valeur vaut $G_n = W^n$, où W est le poids (fixe) entre les sous-buts et n la distance (exprimée en nombre de sous-buts) jusqu'au but (fig. 3).

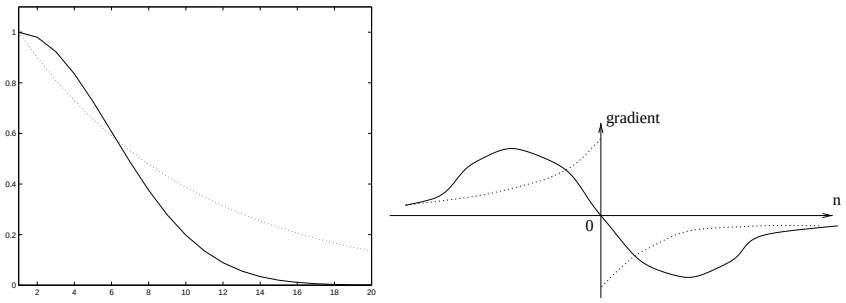

Figure 3. *En haut, H_n (ligne continue) et G_n (ligne pointillé) en fonction n. En bas, gradient de H_n ($\frac{dH_n}{dn}$) et de G_n en fonction de n. Dans le premier cas, le gradient diminue lorsqu'on s'approche du but ($n = 0$), tandis que dans le second cas, ce gradient augmente conduisant à une vitesse non nulle sur le but.*

Comme on peut le voir, quand l'animat se rapproche du but, le gradient (donc $drive$) augmente. La vitesse n'est donc pas nulle quand l'animat atteint

le but, et il peut donc y avoir des oscillations autour de celui-ci avant de l'atteindre (ligne pointillé dans la fig. 4). Pour empêcher ce comportement instable (ou pour s'arrêter sur le but), la vitesse de l'animat doit diminuer lorsqu'il s'approche du but. Nous choisissons alors de calculer le gradient selon l'équation suivante :

$$H_n = exp - \frac{log(\frac{G_n}{W})^2}{2.\sigma^2} \qquad [3]$$

où $\sigma = 5$. La vitesse près du but est maintenant réduite permettant une approche douce (fig. 3) particulièrement intéressante s'il est nécessaire que l'animat s'arrête sur le but.

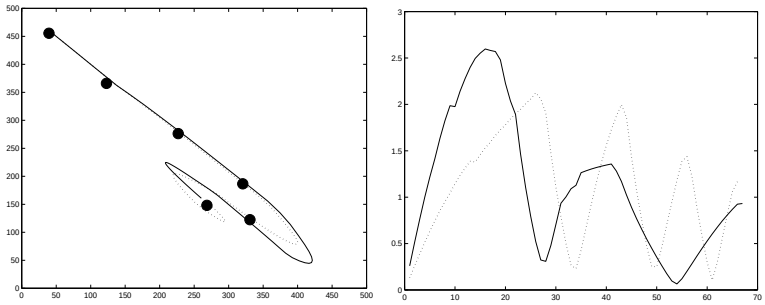

Figure 4. *A gauche : trajectoire d'un animat contrôlé par le gradient de H_n (ligne continue) ou G_n (ligne pointillé). Les gros points sont les sous-buts appris. Les liens les reliant ne sont pas montrés. A droite, courbes de vitesse en fonction du temps (mêmes conventions). La vitesse est plus élevée dans le second cas que dans le premier.*

La comparaison des deux méthodes de calcul du gradient montre que dans le premier cas, l'animat rejoint le but plus rapidement, mais avec une vitesse élevée. En effet, la force d'attraction du but est plus importante de telle sorte que l'animat ne glisse pas autant que dans le second cas. Cependant, il peut arriver que le but ne soit pas rejoint du tout. Dans le second cas, l'animat met plus de temps pour rejoindre le but, mais y arrive avec une vitesse très faible. Cependant, à cause de la force d'attraction plus faible, il peut "graviter" autour avant de l'atteindre. L'interprétation de ces résultats est particulièrement nette dans le cadre des systèmes dynamiques. Ils correspondent alors à l'application des "champs neuronaux" (neural field) sur la carte de planification [AMA 77, SCH 95]. En effet, la notion de distance (exprimée en nombre de sous-buts) peut être rendu continue. On peut alors considérer que le but possède un champ d'activation qui s'étend sur les sous-buts voisins. La valeur de ce champ sur le sous-but n est H_n (fig. 3). Ainsi, de la même façon qu'il est possible de contrôler la direction de déplacement d'un animat en contrôlant sa vitesse angulaire de rotation [BIC 97], nous contrôlons la position de l'animat en modifiant sa

vitesse par l'intermédiaire du gradient de H_n. Dans ce cas là, le but est un point fixe de la dynamique $\dot{n} = F(n)$ (fig. 3). Quand $n = 0$, l'animat est sur le but (et sa vitesse est nulle). De plus, comme ce point fixe est stable, on est sur que l'animat va y converger (théoriquement en un temps asymptotique). Tout ceci n'est plus vrai, si on utilise un gradient calculé à partir de G_n. L'animat peut ne pas converger vers le but, car il n'est plus un point fixe (fig. 3). Il est aussi important de souligner qu'il s'agit d'un processus descendant ("top-down"). En effet, si la carte a d'abord été construite grâce au mécanisme de reconnaissance, maintenant, en retour, elle peut être utilisée pour le contrôle bas niveau de l'animat.

5. Sélection de l'action dans un environnement dynamique

La carte de planification permet déjà de résoudre des problèmes de sélection de l'action dans des environnements statiques [GAU 98]. Cependant, dans un environnement plus réaliste, il faut tenir compte du fait que de nouveaux obstacles peuvent apparaître ou disparaître (une personne qui marche, une porte ouverte ou fermée ...). Dans un premier temps, le changement dans l'environnement pourrait affecter la reconnaissance des lieux. Ce n'est pourtant pas le cas en général, car l'algorithme de reconnaissance de lieux est insensible à la perte (ou l'ajout) d'un certain nombre d'amers. En revanche, la carte de planification peut devenir parfaitement obsolète : certains chemins peuvent disparaître et d'autres peuvent apparaître. Si un obstacle apparaît soudainement, le mécanisme réflexe de suivi peut tout de même permettre d'atteindre le but (voir figure 2). Nous avons testé notre algorithme **tel quel** dans un environnement dynamique (fig. 5). Dans cet environnement, une des deux portes entre les deux pièces peut être fermée. L'animat a appris l'environnement avec la porte ouverte. Quand il y a deux chemins pour aller d'une pièce à l'autre, le chemin direct est préféré, mais l'autre peut aussi être utilisé (figure de gauche). Dans la figure de droite, le chemin direct est fermé. L'animat suit alors l'autre chemin.

Le fait d'avoir continuellement de l'apprentissage dans la carte de planification permet son utilisation dans un environnement changeant. Ainsi de nouveaux buts ou sous-buts sont constamment créés (si de nouveaux lieux sont découverts). Les poids des liens peuvent aussi augmenter si un chemin est utilisé (et augmentent d'autant plus que le chemin est utilisé souvent - figure de gauche). Il y a donc émergence de chemins préférés. Inversement, ces poids diminuent si le chemin n'est pas utilisé (paramètre λ dans l'équation 1). Ainsi, quand la porte est fermée, l'utilisation de la carte de planification conduit l'animat à vouloir traverser le mur. Comme ce n'est pas possible, le mécanisme de suivi d'obstacles va permettre de suivre le mur jusqu'à l'ouverture suivante,

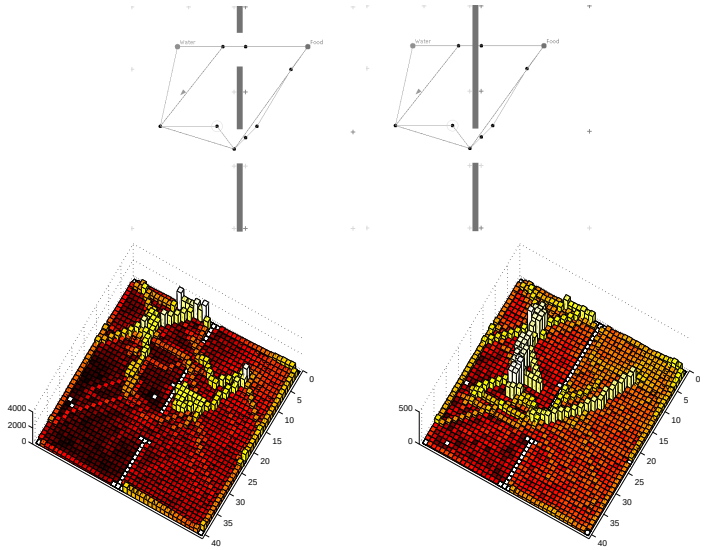

Figure 5. *En haut : carte de planification dans l'environnement avec deux portes (à gauche) et une seule porte (à droite). La carte de planification a été apprise initialement dans le premier environnement. En bas : histogramme de l'occupation de cases élémentaires par l'animat (la hauteur et l'intensité indiquent des endroits très souvent occupés). L'animat est capable de changer sont comportement et de passer par la seule porte ouverte.*

où l'animat reprend alors son chemin vers le but. De cette manière, progressivement, la valeur du lien "traversant" le mur va diminuer (car il n'est plus utilisé, donc plus renforcé), et les liens conduisant vers l'autre ouverture vont être renforcés car effectivement empruntés par l'animat. Il est important de noter qu'après la fermeture de la porte, les amers qui étaient visibles depuis l'autre pièce ne le sont plus. La situation (visuelle) qui a conduit à la création d'un sous-but n'est donc plus valable. Ainsi, dans certains cas, ces sous-buts peuvent ne plus être activés du tout (même si l'animat est exactement à l'endroit auquel ils correspondent) car d'autres sous-buts correspondant au nouvel environnement sont appris (et leur bassin d'attraction s'étend sur ceux déjà appris), ou bien la distance de réponse d'un sous-but déjà appris s'étend. Dans le cas extrême, un sous-but appris dans un environnement peut ne pas être actif du tout dans le nouvel environnement. Il n'est donc plus possible de créer des liens vers lui. Ces liens existent toujours, mais ne peuvent plus être renforcés (et donc leur valeur décroît). Si l'ancien environnement est présenté à nouveau, ces sous-buts sont à nouveau valides, et les liens peuvent être à nouveau utilisés. Comme leur valeur est faible, il faut un certain temps avant qu'ils n'atteignent à nouveau une valeur élevée. Le même processus s'applique aussi pour les liens créés dans le second environnement. De cette manière, plusieurs "couches" de cartes de planification peuvent être stockées en même temps dans la même

structure neuronale. Elles peuvent être liées entre elles à travers certains buts ou sous-buts actifs dans plusieurs environnements. Chaque carte est réactivée par la présentation de l'environnement qu'elle avait appris.

6. Conclusion

La règle d'apprentissage hebbien utilisée est adaptée à la décroissance lente des liens inutilisés, mais ralentit la découverte d'un nouveau chemin (car la décroissance d'un lien qui n'est plus valide est lente). Il serait donc nécessaire d'introduire un terme d'oubli actif pour réagir plus rapidement aux changements de l'environnement.
L'inconvénient majeur de notre algorithme est le calcul du gradient G_n. En effet dans des environnements très grands (avec un grand nombre de sous-buts), le gradient peut devenir très petit. S'il y a un bruit additif, même faible, il deviendra impossible de suivre le gradient. Il apparaît donc nécessaire de combiner plusieurs cartes codant des environnements différents. Il faudra ainsi définir une structure permettant de manipuler des "plans sur des plans". Cette nécessité est aussi renforcée par le fait que pour l'instant la même structure code toutes les cartes.

Ces travaux ont été financés par deux contrats GIS sur les Sciences cognitives intitulés d'une part "comparaison d'architectures de contrôle pour le problème de la sélection de l'action" en collaboration avec l'Animat Lab (J.Y. Donnart, A. Guillot, J.A. Meyer), le LISC (G. Deffuant) et RFIA (F. Alexandre, H. Frezza), et d'autre part "Les robots mobiles comme outils de simulation de la dynamique de modèles neurobiologiques : adaptation et apprentissage dans des tâches de navigation visuelle" en collaboration avec le CRNC (G. Schöner) et le LAAS (R. Chatila).

Références

[AMA 77] S. Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87, 1977.

[ARB 77] M.A. Arbib et I. Lieblich. Motivational learning of spatial behavior. In J. Metzler, ed, *Systems Neuroscience*, pages 221–239. Academic Press, 1977.

[BAC 94] I.A. Bachelder et A.M. Waxman. Mobile robot visual mapping and localization: A view-based neurocomputational architecture that emulates hippocampal place learning. *Neural Networks*, 7(6/7):1083–1099, 1994.

[BEL 58] R.E. Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16:87–90, 1958.

[BIC 97] E. Bicho et G. Schöner. The dynamics approach to autonomous robotics demonstrated on a low-level vehicle platform. *Robotics and Autonomous System*, 21:23–35, 1997.

[BUG 95] G. Bugmann, J.G. Taylor et M.J. Denham. Route finding by neural nets. In J.G. Taylor, ed, *Neural Networks*, pages 217–230, Henley-on-Thames, 1995. Alfred Waller Ltd.

[DON 96] J.Y. Donnart et J.A. Meyer. Learning reactive and planning rules in a motivationnally autonomous animat. *IEEE Transactions on Systems, Man and Cybernetics-Part B*, 26(3):381–395, 1996.

[FRA 98] M.O. Franz, B. Schölkopf, H.A. Mallot et H.H. Bülthoff. Learning view graphs for robot navigation. *Autonomous Robots*, 5:111–125, 1998.

[GAU 95] P. Gaussier et S. Zrehen. Perac: A neural architecture to control artificial animals. *Robotics and Autonomous Systems*, 16(2-4):291–320, 1995.

[GAU 97] P. Gaussier, C. Joulain, S. Zrehen, J.P. Banquet et A. Revel. Visual navigation in an open environment without map. In *International Conference on Intelligent Robots and Systems - IROS'97*, pages 545–550, Grenoble, France, September 1997. IEEE/RSJ.

[GAU 98] P. Gaussier, S. Leprêtre, C. Joulain, A. Revel, M. Quoy et J.P. Banquet. Animal and robot learning: experiments and models about visual navigation. In *Seventh European Workshop on Learning Robots- EWLR'98*, Edinburgh, UK, 1998.

[MEY 91] J.A. Meyer et S.W. Wilson. From animals to animats. In MIT Press, ed, *First International Conference on Simulation of Adaptive Behavior*. Bardford Books, 1991.

[REV 98] A. Revel, P. Gaussier, S. Leprêtre et J.P. Banquet. Planification versus sensory-motor conditioning: what are the issues ? In R. Pfeifer, B. Blumberg, J.A Meyer et S.W. Wilson, eds, *From Animals to Animats : Simulation of Adaptive Behavior SAB'98*, pages 129–138. MIT Press, 1998.

[SCH 92a] N.A. Schmajuk et H.T. Blair. Place learning and the dynamics of spatial navigation:a neural network approach. *Adaptive Behavior*, 1:353–385, 1992.

[SCH 92b] N.A. Schmajuk et A.D. Thieme. Purposive behavior and cognitive mapping: a neural network model. *Biological Cybernetics*, 67:165–174, 1992.

[SCH 95] G. Schöner, M. Dose et C. Engels. Dynamics of behavior: theory and applications for autonomous robot architectures. *Robotics and Autonomous System*, 16(2-4):213–245, December 1995.

[THI 96] C. Thinus-Blanc. *Animal Spatial Navigation*. World Scientific, 1996.

[TOL 48] E.C. Tolman. Cognitive maps in rats and men. *The Psychological Review*, 55(4), 1948.

[TRU 97] O. Trullier, S.I. Wiener, A. Berthoz et J.A. Meyer. Biologically based artificial navigation systems: review and prospects. *Progress in Neurobiology*, 51:483–544, 1997.