DEA IARFA

Évolution de contrôleurs neuronaux pour la commande d'un robot à roues et pattes : vers une tolérance aux pannes.

Thierry HOINVILLE
thoinvil@yahoo.fr

juillet 2002

Stage effectué au LIRIS sous la direction de :

Patrick HÉNAFF
LIRIS, UVSQ – CNRS
Patrick.Henaff@robot.uvsq.fr

Jean-Arcady MEYER
AnimatLab, LIP6 – CNRS
Jean-Arcady.Meyer@lip6.fr

Table des matières

1 Introduction

L'évolution de contrôleurs neuronaux pour la commande de robot n'est pas chose nouvelle. Cependant, la robotique évolutionniste ne s'est pas encore attachée à construire des contrôleurs tolérants aux pannes et adaptatifs aux changements d'environnement. C'est l'objectif du projet IRON qui vise à accroître l'autonomie et la capacité d'adaptation des robots. Ceci, en confrontant des architectures de contrôle biomimétique ajustées par évolution à des perturbations de toutes sortes : rupture d'articulation, usures, obstacles, pertes d'adhérence, etc.

Dans un cadre plus restreint, notre travail consiste à élaborer par évolution un contrôleur neuronal capable de basculer d'un comportement à un autre suite à une perturbation brutale. Cette étude est appliquée à la commande d'un robot à roues et pattes conçu au LIRIS.

L'intérêt scientifique de ce travail est principalement d'ouvrir la voie aux développements avenirs qui seront engagés dans le projet IRON. Par ailleurs, comme il sera évoqué plus loin, il s'agit de tester des méthodes simples et éprouvées afin de se forger une base de comparaison.

Nous commencerons par présenter le robot sur lequel se base notre travail dans la section suivante. Notre méthode évolutionniste est détaillée dans la section 3. Nos expériences et résultats font l'objet de la section 4. Une discussion de notre travail est ensuite entreprise, avant de conclure, dans la section 5.

2 Matériel

Notre étude expérimentale se base sur un robot réel appelé **Zibana**[1] développé au LIRIS par E. Monacelli [Ben Salem et al., 2002]. La locomotion du robot est *hybride* : il possède des roues et des pattes. Cette caractéristique lui permet de s'adapter à son environnement. Ainsi, il peut avoir recours à ses roues sur un sol plat pour économiser de l'énergie et accroître sa vitesse, mais si le sol devient accidenté, la locomotion à l'aide des pattes est préférée pour sa capacité de franchissement. Cette double locomotion accroît la robustesse du robot. En effet, si l'une est affectée d'un dysfonctionnement, l'autre peut assurer la réussite de la mission du robot[2].

La morphologie des pattes de Zibana est un standard de la robotique : il s'agit de *bras manipulateurs* sans poignets. Les segments d'un bras sont

[1]Nom d'un désert algérien et «balance» en sumérien.

[2]Dans le cadre de l'approche Animat, la mission du robot consiste essentiellement à assurer sa pérénité.

assemblés par trois *liaisons rotoïdes*, deux pour l'épaule et une pour le coude (figure 1). Les trois degrés de liberté de cette structure permettent de positionner l'extrémité de la patte à n'importe quel point de l'espace tridimensionnel atteignable (i.e. limité par les butées articulaires et les dimensions des segments).

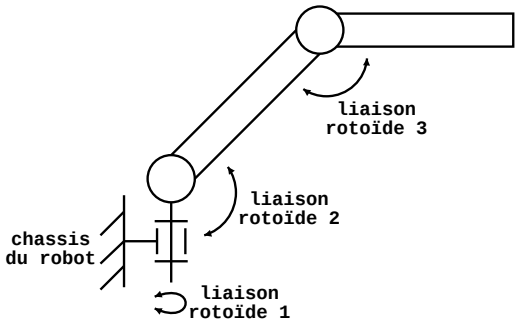


FIG. 1 – Schéma mécanique d'un bras manipulateur sans poignet.

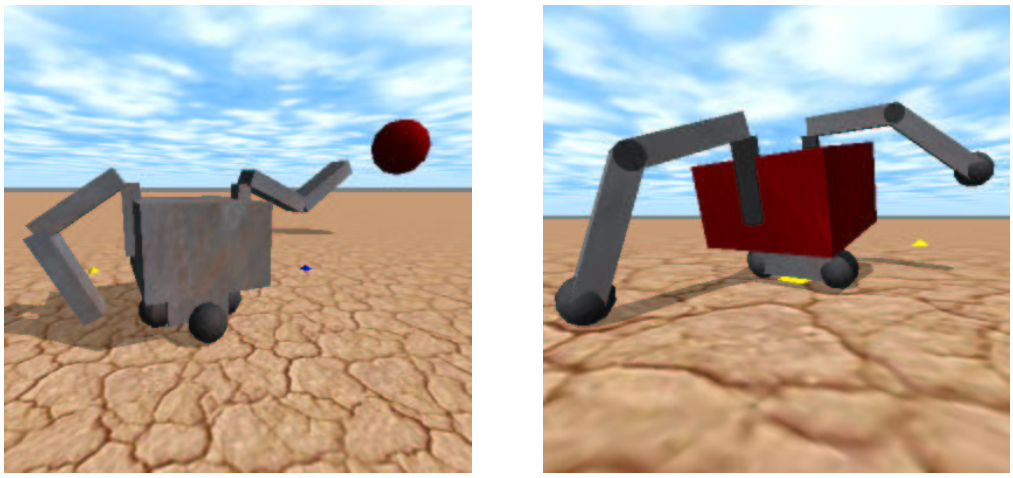
La réalisation physique du robot n'étant pas finalisée, nous avons envisagé lors de nos expériences en simulation deux dispositions spatiales des roues :

- La première configuration fait intervenir trois roues. Deux roues motrices sont placées côte à côte de manière à ce que leur axe de rotation soient confondus. Cette configuration a l'avantage d'autoriser le changement de direction sans recourir à des manœuvres compliquées en appliquant un différentiel de vitesse sur les roues. Une roue libre[3] est placée en face avant du robot. Le triangle de sustentation ainsi formé permet d'assurer une stabilité statique (figure 2(a)).
- Dans la seconde configuration, seules deux roues sont employées. Celles-ci sont disposées en ligne à la manière d'un roller. La roue avant est motrice. Dans ce cas, il n'y a plus de stabilité statique et les changements de direction doivent être entrepris à l'aide des bras en appliquant un effort sur le sol (figure 2(b)).

Il faut noter que dans les deux cas le contrôleur doit assurer la *stabilité dynamique* du robot en actionnant ses pattes. Cette constatation, évidente pour la configuration en ligne des roues, est aussi valable lorsque le robot repose sur trois roues. En effet, le triangle de sustentation est excentré par rapport au centre de gravité de l'ensemble. Une accélération trop forte vers l'avant ou une pente trop importante suffirait alors à lui faire perdre l'équilibre et basculer sur le dos. Par ces morphologies volontairement instables nous favorisons la double utilisation des pattes : **propulsion** et **équilibre**. Lorsque

[3]Une liaison sphèrique autorise deux axes de rotation.

l'environnement le permettra le robot utilisera ses roues pour se mouvoir et gardera son équilibre en actionnant ses bras, tandis que si le sol devient accidenté ou si une défaillance survient, ses roues ne lui seront d'aucun secours et il actionnera ses bras comme des pattes pour continuer son déplacement. Le contrôleur doit donc être capable d'adapter son fonctionnement en *basculant* de la phase de maintien de l'équilibre à la phase de propulsion. L'adaptation aux pannes brutales dont fait l'objet le projet IRON nécessitera de telles transitions de phases.



(a) Roues coaxiales (b) Roues en ligne

FIG. 2 – Les deux configurations morphologiques

2.1 Perceptions

Afin que Zibana puisse réaliser sa tâche consistant essentiellement à se déplacer, il lui faut percevoir son environnement. Pour ce faire, nous l'avons doté d'un capteur de contact binaire à l'extrémité de chaque patte. Comme nous le préciserons par la suite, lors de notre première expérience le robot devait atteindre une position que nous déterminions comme son but, en robotique on parle d'**asservissement** en position. Ainsi, nous avons incorporé sur le robot simulé un capteur donnant la distance et le cap de cet objectif.

De par ses bras, notre animat bénéficie d'un certain potentiel de *reconfiguration spatiale*, il peut modeler sa morphologie en vue de s'adapter à son environnement. Par conséquent, si l'on veut qu'il soit en mesure d'utiliser cette propriété, des capteurs proprioceptifs sont indispensables. Le plus

simple est de fournir au contrôleur les positions angulaires des articulations des pattes et éventuellement les vitesses de rotation.

Du fait de son instabilité dynamique, il faut doter le robot d'un inclinomètre donnant le tangage et le roulis de son chassis. Enfin, un capteur de vitesse globale du robot nous a paru essentiel pour détecter la perturbation que nous lui faisont subir au cours d'une de nos expériences et ainsi provoquer le changement de phase. Concernant le robot réel, l'inclinaison et la vitesse de la plateforme seront calculées à partir des données d'un accéléromètre.

2.2 Actions

Notre robot comporte des servomoteurs pour les articulations des bras et des moteurs à courant continu pour les roues. Le logiciel de simulation que nous employons est capable de motoriser chaque liaison entre deux corps. Il suffit de paramétrer le lien par la valeur d'un couple mécanique maximum correspondant à la caractéristique du moteur. La commande du moteur se fait en lui appliquant une vitesse de rotation. En réalité, la commande par position angulaire est plus courante mais un simple calcul permet de passer de l'une à l'autre. Le mouvement du point terminal d'un bras est donc commandé par trois vitesses angulaires.

3 Méthode

Notre méthodologie se situe dans le cadre de la **robotique évolutionniste**. Cette discipline récente confie l'élaboration du contrôleur ou de la morphologie d'un robot (et parfois des deux) à un processus évolutif inspiré de la théorie de l'évolution des espèces proposée par Darwin. Parmi les nombreuses méthodes de la robotique évolutionniste, nous avons employé celle qui consiste à faire évoluer les poids synaptiques d'un **réseau de neurones** (section 3.1) par un **algorithme génétique** (section 3.2). Le calcul des performances de chaque réseau est réalisé par l'évaluation en simulation d'un robot dans lequel il s'*incarne*. Le choix de cette méthode provient d'abord du projet IRON dans lequel s'insère notre travail. De plus, ceci permet d'évaluer les limites de ces outils et par conséquent oriente l'élaboration de techniques plus avancées. Enfin et surtout, cette étude permet de se donner un point de référence pour comparer les différentes solutions, une comparaison qui fait souvent défaut dans l'approche Animat [Guillot and Meyer, 2000].

3.1 Le contrôleur neuronal

Le modèle de neurone implanté dans notre contrôleur est le modèle formel traditionnel du connexionnisme. Son fonctionnement est une approximation grossière des neurones naturels. Malgré cela, de nombreuses études théoriques et empiriques ont montré que des réseaux de tels neurones sont capables d'effectuer des tâches assez complexes. Les entrées d'un neurone sont pondérées par les poids synaptiques des connexions puis sommées et le résultat de cette somme est injecté dans une fonction d'activation[4] donnant l'état d'excitation ou d'inhibition du neurone.

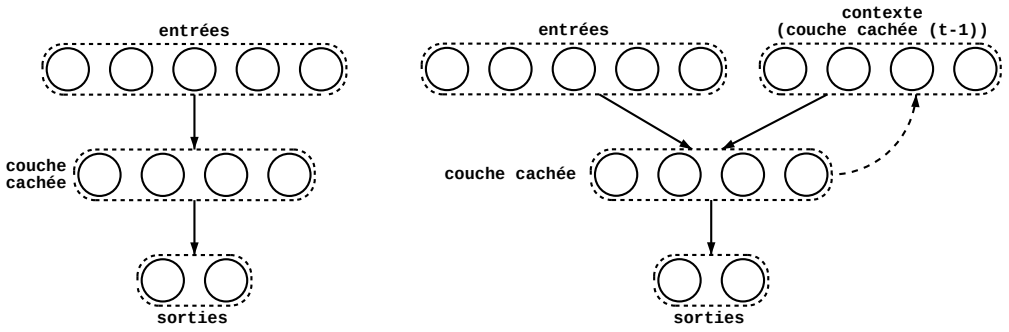
Dans nos expériences, nous avons testé deux architectures de réseaux :

- La première est celle du **perceptron multi-couches** (figure 3(a)). Nous utilisons une seule couche cachée entièrement reliée à la couche d'entrée et à la couche de sortie. Ce modèle ne possède pas d'état interne, les mêmes entrées aboutiront aux mêmes sorties. Ces réseaux sans mémoire sont néanmoins susceptibles de commander des robots réels dont le comportement dynamique est important [Hénaff and Delaplace, 1996, Hénaff and Chocron, 2002].
- La seconde est celle du **réseau récurrent** de Meeden (figure 3(b)) qui permet théoriquement l'apparition de comportements plus complexes. Cette architecture est en fait un perceptron standard auquel on ajoute des connexions récurrentes qui agissent comme un contexte au traitement des entrées : les activations de la couche cachée à l'instant t sont réinjectées en entrée du réseau à l'instant $t+1$. Cela permet de garder une mémoire de l'état précédent, deux mêmes entrées pourront ainsi donner des sorties différentes. Bon nombre d'articles décrivent cette architecture comme celle donnant les meilleurs résultats avec ce modèle de neurone pour la commande de robot [Meeden et al., 1993, Ziemke, 2000].

3.2 L'algorithme génétique

Comme il a été présenté plus haut, un algorithme génétique est chargé de construire des contrôleurs **adaptés** aux épreuves subies par le robot. A cette fin, le phénotype d'un individu, c'est à dire le réseau de neurones, doit être encodé sous forme d'un génotype sur lequel des opérateurs génétiques s'appliquent pour donner de nouveaux individus. Pour simplifier, nous figeons la structure du réseau en fixant arbitrairement le nombre de neurones cachés. Un réseau est codé sous forme d'un *vecteur de réels* correspondant aux efficacités synaptiques des connexions et aux valeurs des biais des fonctions

[4] Nous utilisons une sigmoïde



(a) Perceptron

(b) Réseau récurrent de Meeden

FIG. 3 – Architectures des contrôleurs neuronaux expérimentés. Les flèches solides représentent une couche synaptique connectant entièrement les couches neuronales. La flèche en pointillés représente une simple copie de neurone à neurone entre couches.

d'activation de chaque neurone. Ce codage est plus naturel pour notre problème que le codage binaire qui nécessiterai un temps de calcul important encore alourdit par l'emploi de codage de Gray.

L'opérateur de recombinaison génétique est un simple **crossover à deux points**. Il a l'avantage d'autoriser un plus grand nombre de combinaisons que le crossover à un point qui n'est en fait qu'un cas particulier où le deuxième point est fixe. En outre, nous le préférons au crossover uniforme pourtant plus innovant[5] mais surtout plus destructeur comme l'explique la théorie des schémas [Holland, 1975, Goldberg, 1989].

Le second opérateur est la mutation. Celui-ci pose un problème majeur du fait de notre codage. Il est souvent admis que le codage binaire amène de meilleures performances que le codage réel. Cependant, cette différence est certainement provoquée par la complexité et la diversité des mutations sur des nombres réels. En effet, la mutation en binaire s'obtient par le simple basculement d'un bit, tandis que la mutation de gènes réels appelle généralement la génération d'un bruit gaussien contenant son lot de paramètres à ajuster. Afin de remédier à ce problème, nous employons un opérateur de **mutation dynamique** [Janikow and Michalewicz, 1991]. La loi de probabilité dont est issue la valeur de la mutation est fonction du nombre courant de générations. Cette mutation favorise l'exploration en début d'évolution et l'exploitation en fin d'évolution. Elle est définit comme suit :

Soit :

[5]Toutes les combinaisons alléliques sont possibles.

- v_k, la valeur du k-ième gène du chromosome à muter ;
- v'_k, la valeur mutée de ce gène ;
- $[MIN, MAX]$, le domaine de valeurs des gènes ;
- *bool*, un booléen aléatoire ;
- r, un reél aléatoire (loi uniforme sur $[0, 1]$) ;
- g, le nombre courant de générations ;
- et G, le nombre total de générations.

$$v'_k = \begin{cases} v_k + \Delta(g, MAX - v_k) & \text{si} \quad bool = 0 \\ v_k - \Delta(g, v_k - MIN) & \text{si} \quad bool = 1 \end{cases}$$

$$\text{où } \Delta(g, y) = y\ (1 - r^{(1-\frac{g}{G})^B})$$

La constante B représente la vitesse de passage de la phase d'exploration à la phase d'exploitation. Nous avons choisit la même valeur que Janikow ($B = 5$).

Nos premières expériences nous ont rapidement confronté au phénomène de **convergence prématurée** donnant des solutions d'optima locaux non satisfaisantes. Au delà de la robotique évolutionniste, c'est un problème majeur des techniques évolutionnistes. Sa résolution passe avant tout par des améliorations de la *méthode de sélection*.

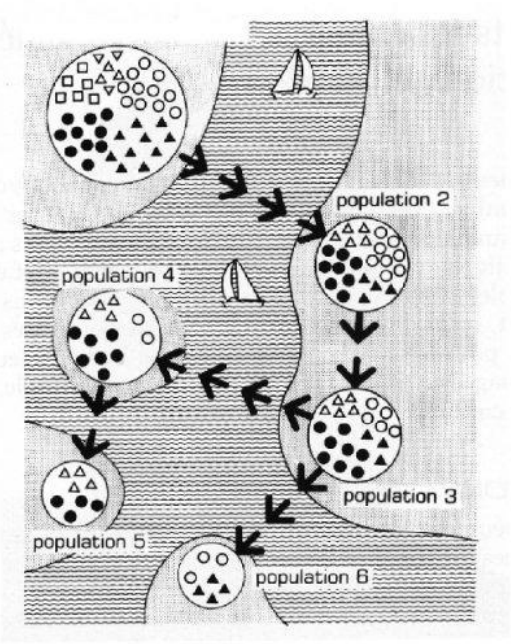
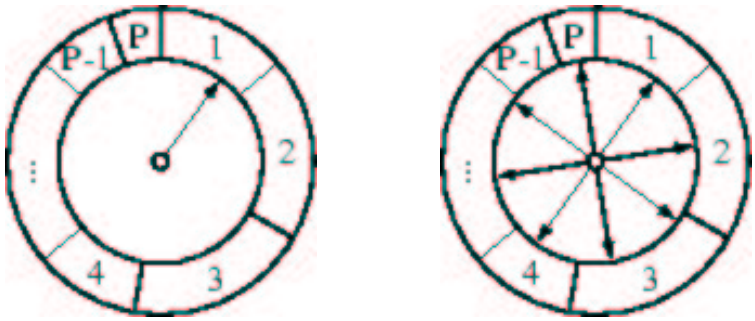


FIG. 4 – La dérive génétique : des échantillonnages successifs conduisent parfois à la disparition de certains allèles. Schéma issu de [Langaney, 1988]

L'*échantillonnage* des individus admis à se reproduire est un point très important. Ainsi, la technique basique de roue de loterie conduit généralement à une **dérive génétique** (figure 4). La dérive génétique est définie par l'accumulation d'*erreurs stochastiques* entrainant un apauvrissement allélique de la population. En pratique, on observe au bout de seulement quelques générations qu'un seul génotype a colonisé la population entière. Un indicateur simple de la diversité dans la population est la *variance* de la fonction d'adaptation (*fitness*) [Rogers and Prügel-Bennett, 1999]. Il faut cependant noter que la convergence d'un algorithme génétique aboutira invariablement à une relative baisse de cette variance. De même, si aucune technique de préservation des niches écologiques n'est utilisée, la diversité des individus de la population aura tendance à régresser puisque la taille de la population est fixe. Il y a convergence prématurée lorsque l'aspect stochastique de l'échantillonnage prend le pas sur l'évaluation des performances de chaque individu. L'échantillonnage que nous utilisons pour notre sélection est le **Stochastic Universal Sampling** (SUS) [Baker, 1987]. Le SUS est une technique optimale garantissant que chaque individu ne sera ni sur-selectionné, ni totalement ignoré. Contrairement à la roue de loterie traditionnelle (figure 5(a)) qui est actionnées p fois pour p individus à la génération suivante, le SUS utilise une seule fois la roue en répartissant uniformément p pointeurs sur sa circonférence (figure 5(b)).



(a) Roue classique (b) Roue de SUS

FIG. 5 – Roues de loterie utilisées pour échantillonner la population durant la phase de sélection.

Par ailleurs, une *modification de la fonction d'adaptation* permet également de se prémunir contre les convergences vers des optima locaux [Goldberg, 1989]. Comme l'a souligné Goldberg, les diverses méthodes d'altération de la fitness ont un caractère *ad hoc*, il convient de les tester afin de déterminer

lesquelles sont les plus adaptées au problème que l'on doit résoudre. Nos tests ont porté sur le changement d'échelle linéaire ($fitness_{mod} = afitness+b$), exponentiel ($fitness_{mod} = e^{-fitness}$) et sur la **méthode du rang** ($fitness_{mod} = f(rang)$) [Baker, 1985]. Nous avons opté pour cette dernière en raison de sa simplicité et de son efficacité relative à notre cas.

Précisons que nous avons adopté l'algorithme *générationnel* standard où chaque génération succède entièrement à sa précédente. En effet, plusieurs articles montrent qu'un tel recouvrement (*generation gap*) contribue à éviter la dérive génétique [De Jong and Sarma, 1993, Rogers and Prügel-Bennett, 1999] et nécessite un nombre d'individus moindre que les algorithmes à faible recouvrement (*steady-state*).

Enfin, le tableau 1 résume nos choix quantitatifs.

Paramètre	**Valeur**
Nombre de générations	1000
Nombre d'individus	100
Probabilité de crossover	0.8
Probabilité de mutation	10^{-2}
Pression de sélection	1.5

TAB. 1 – Valeurs des paramètres de l'algorithme génétique

3.3 Le simulateur

Le dénominateur commun des différentes approches de la robotique évolutionniste est le besoin de recourir à un simulateur. En effet, toute évolution artificielle demande une *évaluation* de chaque individu durant l'ensemble des générations afin de pouvoir procéder à la sélection des meilleurs d'entre eux, lesquels pourront ensuite se reproduire. Comme l'évolution biologique, ce grand nombre d'évaluations demanderai un temps extrèmement long si elle devait se dérouler dans l'environnement réel. Beaucoup de simulateurs physiques existent sur le marché, chacun comporte ses avantages et ses inconvénients. Il convient de bien étudier leurs spécificités afin de dégager le meilleur candidat en fonction de nos besoins. C'est le travail qu'a réalisé G. Bayard au début de sa thèse. Le choix qu'il préconise se porte sur le simulateur **Open Dynamic Engine**[6] (ODE). En premier lieu, c'est un *simulateur dynamique*. La prise en compte de la dynamique des systèmes mécaniques nous

[6] www.q12.org

est indispensable car elle joue un grand rôle dans les locomotions par appuis discontinus comme c'est le cas en robotique à pattes. En outre, ODE bénéficie d'une grande stabilité et d'une rapidité excellente (20 fois supérieure au temps réel), avantages cruciaux puisque chaque évolution demande un nombre énorme d'évaluations avant d'obtenir une solution. En revanche, ce simulateur privilégie la vitesse à la précision (sans toutefois la compromettre). Cette différence quantitative n'est pas fondamentale pour le succès de l'approche évolutionniste car il est beaucoup plus important que le simulateur soit juste sur le plan qualitatif. Une autre raison de ne pas redouter ce manque possible de précision est que notre but est justement de tester l'adaptation de notre animat.

4 Expériences et résultats

Notre étude s'est articulée autour de deux expériences complémentaires du point de vue d'une part, de la configuration morphologique du robot et d'autre part, de l'approche employée.

4.1 Robot à trois roues et approche naïve

4.1.1 Dispositif expérimental

Nos premiers essais concernent la version à trois roues du robot (voir l'image 2(a) page 3). La mission assignée au robot est de rejoindre un point donné du plan. La fitness retenue est la distance finale[7] à ce point. Dans un souci de généralisation, l'orientation initiale du robot est aléatoire. Les perceptions externes du robot se composent des capteurs de contact binaire et du senseur lui indiquant le cap et la distance de l'objectif introduits plus haut. Ses entrées proprioceptives sont les inclinaisons de son corps ainsi que la configuration angulaire et cinématique de ses articulations. Un seul perceptron (figure 6) dispose de toutes ces données pour commander l'ensemble de la locomotion du robot, c'est à dire pour calculer la vitesse angulaire de chaque articulation des bras et des deux roues motrices. L'environnement se limite à une surface plane qui est soit vierge de tout obstacle, soit agrémenté d'un plan incliné perturbant la trajectoire et l'équilibre du robot.

4.1.2 Résultats

Plusieurs dizaines d'évolutions ont été lancées dans le cadre de ce dispositif. Leurs résultats sont pour la plupart non satisfaisants, faisant apparaître

[7] Chaque évaluation de cette fitness nécessite une simulation de 1000 cycles.

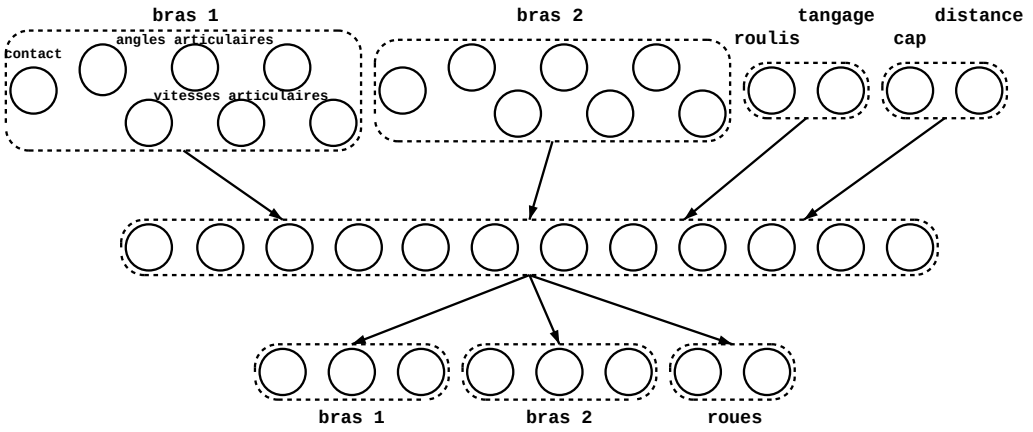


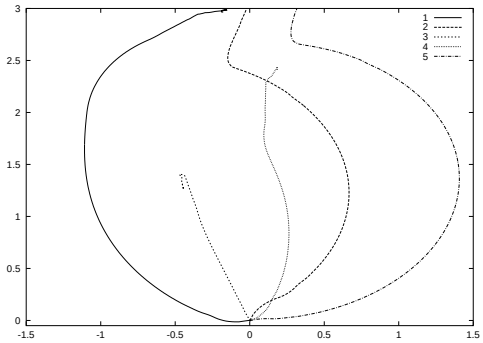
FIG. 6 – Schéma descriptif du contrôleur neuronal

des comportements assez chaotiques. Ainsi, des corrélations sensori-motrices et motrices inapropriées générant des mouvements erratiques affectent beaucoup de contrôleurs. Un exemple frappant est un changement de l'allure d'une roue impliquant une inutile reconfiguration articulaire d'un bras. Ce problème est symptomatique de l'architecture entièrement connectée de notre réseau. En tous cas, nous n'avons obtenu aucun individu s'équilibrant à l'aide de ses bras, même en présence du plan incliné.

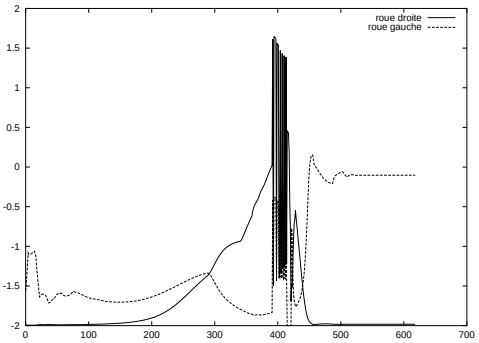
En revanche, quelques évolutions ont abouties à un même comportement de robot réalisant la tâche de positionnement sans utilisation des bras. Dans un premier temps, la stratégie du contrôleur consiste à bloquer l'ensemble des articulations en butées de telle façon que les bras n'entravent pas la progression de l'animat en touchant le sol. Dans un second temps, un différentiel de vitesse est appliqué aux moteurs des roues afin que l'animat s'oriente vers la cible. Le déplacement se fait vers l'arrière, la roue libre à l'avant évite la chute durant la phase d'approche. Une simple perturbation de cette allure est générée et suffit à provoquer la chute et ainsi l'arrêt du robot lorsque l'objectif est atteint (figure 7(b)). En outre, quelque soit son orientation initiale, l'animat est capable se réorienter vers le point désiré, révélant ainsi une généralisation (figure 7(a)).

4.1.3 Vers une autre expérience

Ne répondant pas à nos attentes, ces résultats nous ont amené à discuter de notre démarche expérimentale. Tout d'abord, le mélange de comportements de haut niveau (asservissement) et de bas niveau (déplacement, maintien de l'équilibre) est trop complexe pour être obtenu en une seule étape. Ensuite, il est préférable de séparer le contrôle des roues de celui des bras,



(a) Trajectoires obtenues en faisant varier l'orientation initiale de l'animat (entre $-\pi$ et π). Le robot est placé à t_0 au point $(0, 0)$ et son but est de rejoindre le point $(0, 3)$.



(b) Commande des roues correspondant à la quatrième trajectoire. Le robot atteint la zone voulue vers t_{400} et déclenche sa chute en perturbant la commande des roues par des accélérations brutales.

cette indépendance autorisant une plus grande robustesse. En effet, au cas où un module subirait une avarie l'autre le supplérait pour que l'animat survive. Enfin, il apparaît que la morphologie du robot dans sa version à trois roues est incohérente, car du fait de leur disposition latérale, les bras sont d'une efficacité moindre à redresser le robot lorsqu'il chute en arrière. Par conséquent, les roues en ligne sont plus adéquates car elles entrainent des chutes sur les flancs : là où les bras disposent de plus de liberté. Or, comme l'a montré Pfeifer, la morphologie d'un animat est aussi importante que son contrôleur et permet parfois de largement le simplifier [Pfeifer, 2000].

4.2 Roues en ligne et approche pragmatique

4.2.1 Dispositif expérimental

Les considérations précédentes ont orienté l'élaboration de notre deuxième expérience. Nous avons choisit la morphologie des roues en ligne car elle impose et facilite l'emploi des bras pour le contrôle de l'équilibre. D'autre part, nous avons simplifié l'objectif : la mission du robot est de se déplacer, la fitness est la *distance parcourue dans une direction* donnée pendant le temps de simulation. En effet, s'il est indispensable pour rendre le robot fonctionnel, l'asservissement en position n'est pas nécessaire dans le cadre de notre problématique. De plus, l'ajout d'un module s'en chargeant est tout à fait envisageable, cette technique *incrémentale* où le développement est assumé par le concepteur a déjà donné de bons résultats sur des robots hexapodes

[Filliat, 1999]. L'évolution porte ici sur un réseau contrôlant uniquement les pattes. La roue motrice est contrôlée par nos soins. Les informations perceptives du réseau sont les positions et vitesses angulaires de chaque articulation, les indicateurs de contacts, le roulis, la vitesse de roulis et la vitesse du chassis (figure 7).

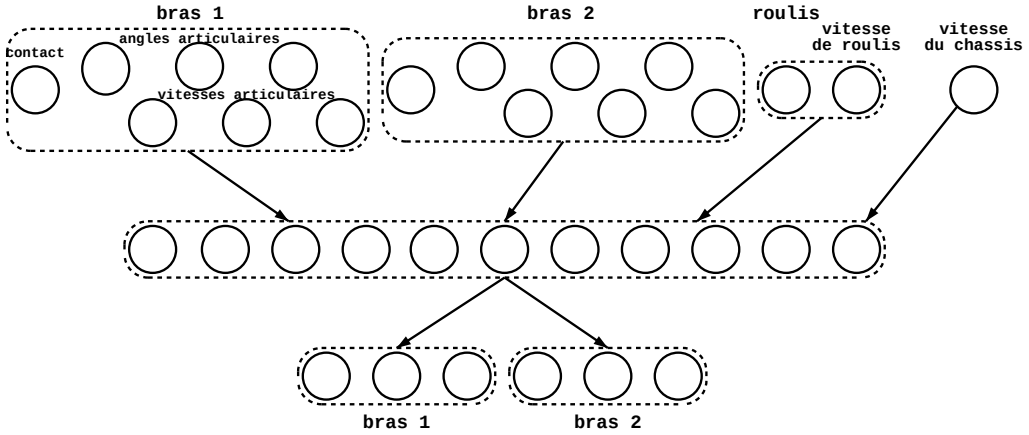


FIG. 7 – Schéma descriptif du contrôleur neuronal

4.2.2 Résultats sans perturbation

Nous appliquons une vitesse de rotation constante à la roue motrice. Dès lors, le meilleur moyen d'aller le plus loin possible pour le robot est de maintenir son équilibre en s'aidant de ses pattes. L'évolution a apporté des résultats satisfaisants : avec un simple perceptron, des mouvements *périodiques* sont apparus aux articulations de chaque bras (figures 8). Ainsi, le robot maintient son équilibre par appuis successifs des extrémités des pattes. Néanmoins, l'ensemble de ces résultats est issu d'une configuration où la gravité, suite à une erreur d'échelle, est très faible[8]. Des valeurs plus réalistes n'ont pour l'instant abouties à aucun résultat du même type, qu'il s'agisse d'un perceptron ou d'un réseau récurrent.

4.2.3 Résultats avec perturbation

Afin d'élaborer un contrôleur capable de basculer d'un fonctionnement à un autre pour s'adapter aux évènements qu'il subit, nous avons repris la configuration de faible gravité en y incorporant une *perturbation*. Lorsque le robot franchi une certaine distance, nous transformons la roue motrice en

[8]Un cinquantième de la valeur terrestre.

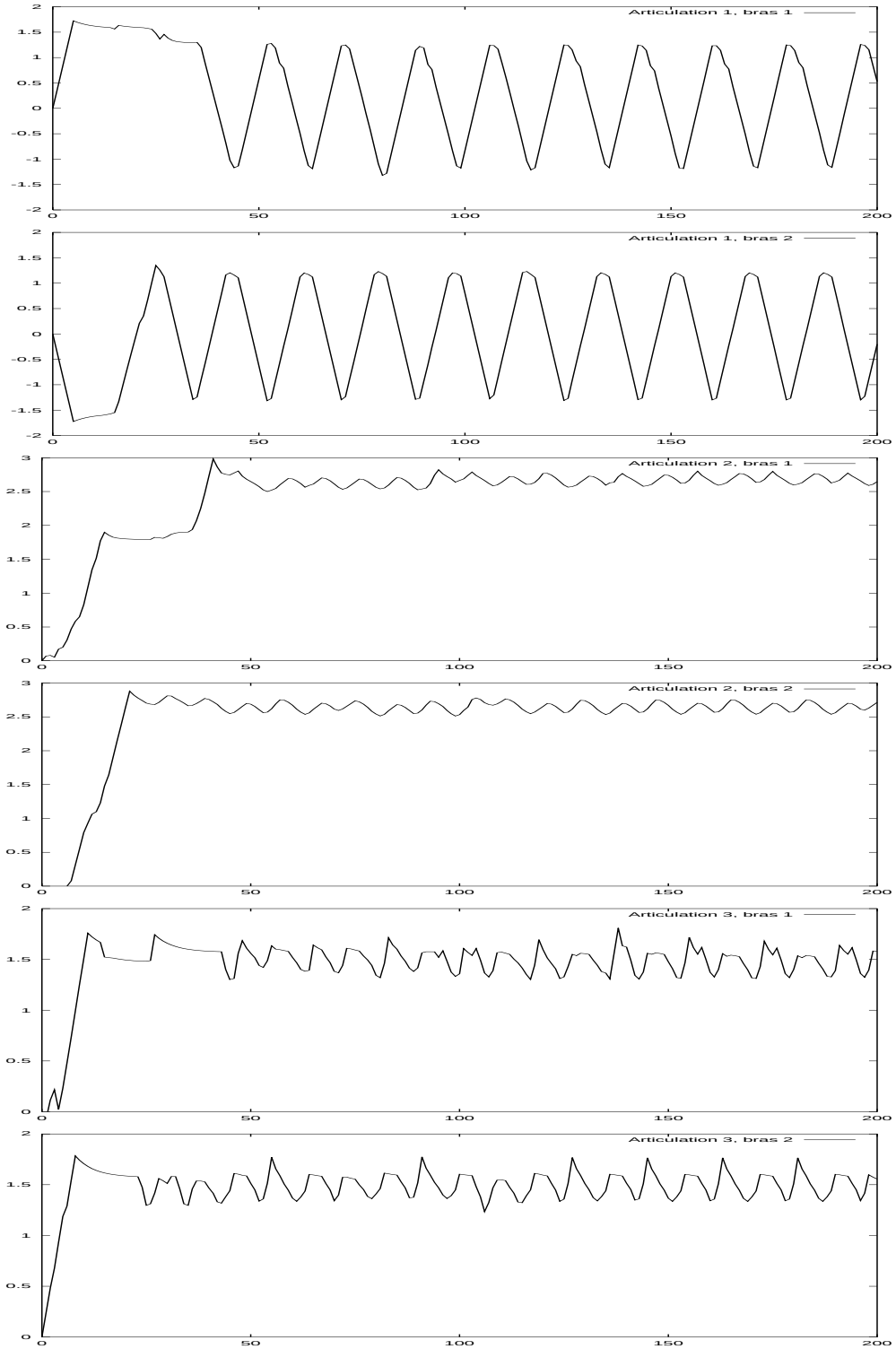


FIG. 8 – Mouvement périodique. (Positions angulaires de chaque articulation au cours du temps)

roue libre, émulant ainsi un dysfonctionnement de type rupture de transmission mécanique. Ainsi, le robot doit dans un premier temps s'équilibrer puis détecter que sa vitesse diminue[9] et enfin passer à la locomotion à patte afin de continuer sa progression.

Il est apparu un individu exhibant deux comportements successifs distincts. Il commence son parcours en imposant à ses bras un mouvement de faible amplitude mais saccadé lui permettant de tenir son équilibre assez longtemps. Puis, un peu avant que la perturbation n'ait lieu, il change de comportement en générant un mouvement de forte amplitude assez chaotique lui permettant par appuis successifs d'une patte sur le sol d'aller le plus loin possible (figure 9). La panne survenant toujours au même endroit, l'évolution a conduit à un individu spécialisé qui *anticipe* l'évènement.

5 Discussion

5.1 La méthode

Comme il est dit plus haut, notre méthode est une des plus simples de la robotique évolutionniste. Elle est ici confrontée à des situations expérimentales complexes qui permettent d'en dégager les limites. Pour cela, nous employons des fitness les plus endogènes possible. De plus, ces fitness non biaisées ont l'avantage de ne pas canaliser l'évolution vers des solutions préétablies par le concepteur [Filliat, 1999]. Il ressort des résultats de notre première expérience que l'évolution seule n'est pas capable d'aboutir à des comportements complexes. Le développement et l'apprentissage sont primordiaux. Une démarche incrémentale consistant à construire le contrôleur module par module revient à combiner l'évolution à un développement totalement assumé par nos soins. Ainsi, l'évolution d'un module spécialisé lors de notre deuxième expérience donne de meilleurs résultats.

5.2 Allures périodiques

La seconde expérience montre que l'évolution d'une simple architecture *feed-forward* permet l'apparition d'allures périodiques assurant le maintien de l'équilibre du robot même en utilisant une fitness ne favorisant pas explicitement ce type de comportement. Toutefois, certains individus génèrent cette périodicité en usant du fait que ODE gère mal les butées articulaires en particulier lorsque le pas de temps est relativement grand. En effet, un moteur s'opposant à une butée aura tendance à faire osciller l'articulation

[9] Il y a un frottement visqueux dans les roulements.

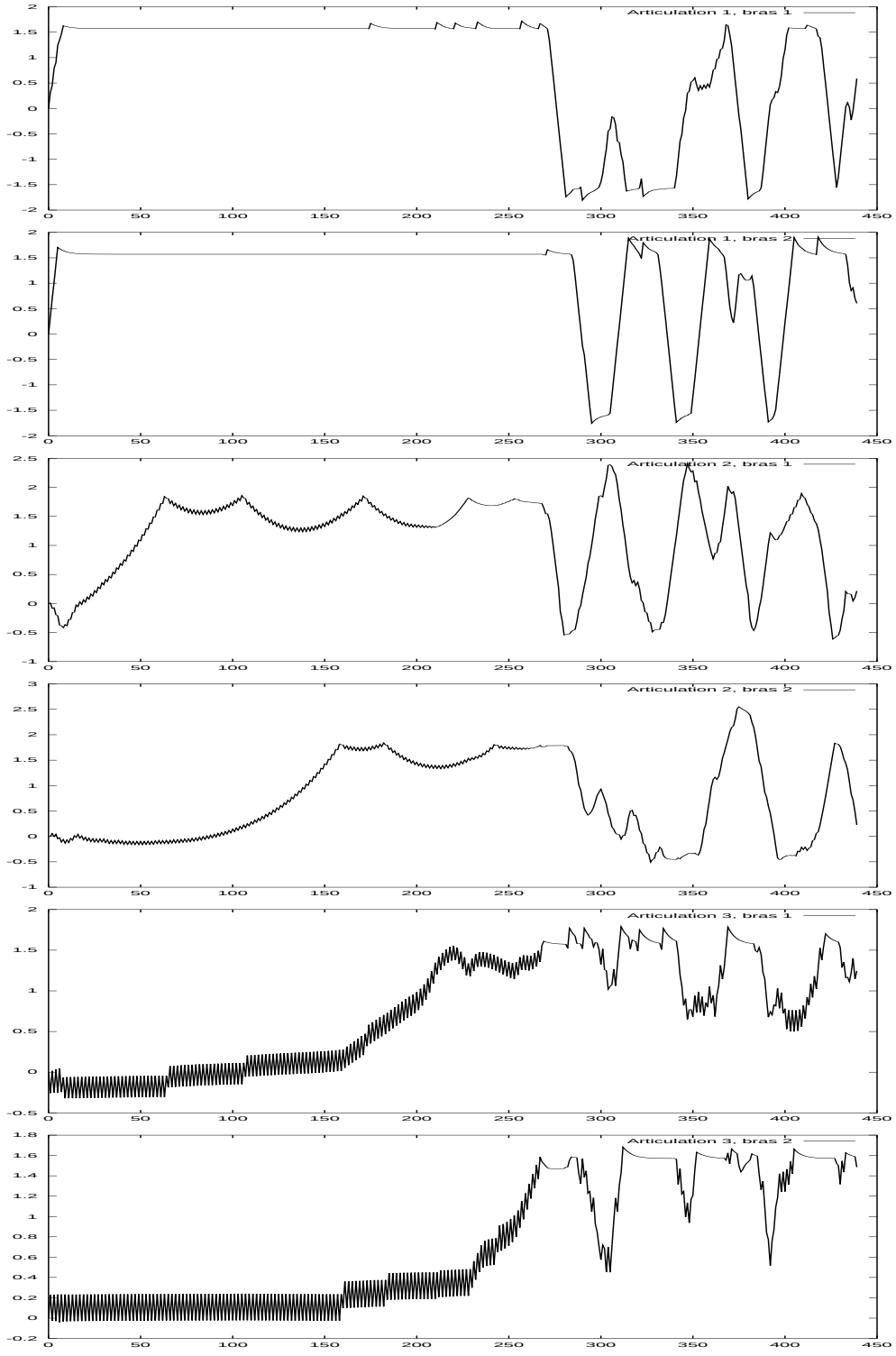


FIG. 9 – L'évolution dans un environnement perturbé aboutie à un comportement composé de deux phases distinctes. La perturbation à lieu à t_{282} (Positions angulaires de chaque articulation au cours du temps)

autour de cette limite. Ainsi, les mouvements périodiques des deuxièmes et troisièmes articulations de chaque bras sur la figure 8 sont issus de tels phénomènes. Ceci renforce l'idée que l'aspect qualitatif du simulateur est très important afin que l'évolution ne soit pas parasitée par des artefacts. ODE devra donc être validé par un mécanicien. Une implication du LIRIS dans le développement de ce simulateur serait bénéfique et facilitée par son statut *opensource*.

5.3 Faible gravité

Les résultats les plus intéressants étant issus d'évolutions dans un environnement à très faible gravité, il convient d'analyser les effets dynamiques d'une pesanteur négligeable. Formellement, la dynamique d'un système articulaire peut se résumer par l'équation :

$$M.\ddot{q} + S(q, \dot{q}) + G(q) = \tau + \Delta^T.F_{ext}$$

Le terme de Coriolis ($S(q, \dot{q})$) est négligeable car dans notre cas les vitesses de rotations ne sont pas assez importantes pour entrainer de tels phénomènes. Le contrôleur agit directement sur le terme moteur (τ) pour s'opposer aux contraintes de gravité ($G(q)$), de contact ($\Delta^T.F_{ext}$) et en partie à celle de l'inertie ($M.\ddot{q}$). Lorsque la pesanteur devient négligeable, on voit facilement qu'une contrainte disparait. Cette contrainte est sans doutes la plus importante et sa disparition facilite la tâche du contrôleur, ce qui explique que l'algorithme génétique aboutisse à de meilleurs résultats dans cette situation.

Par ailleurs, une autre raison pourrait être un problème de type *bootstrap*. En effet, en début d'évolution, la gravité réelle ne permet peut-être pas, du fait de leur égale médiocrité, de départager les contrôleurs. Une solution serait d'appliquer une gravité progressive durant l'évolution.

5.4 Changement de phase

L'ajout d'une perturbation de type panne dans la seconde expérience a conduit l'évolution à trouver un contrôleur capable de générer deux comportements distincts, répondant ainsi à la problématique que l'on s'est fixée. Une étude superficielle du contrôleur semble montrer que le réseau de neurones actionne une articulation jusqu'à une certaine position angulaire qui déclenche la transition brutale de comportement. Le contrôleur du robot se sert donc de la configuration morphologique du robot comme d'une mémoire lui permettant de repérer le moment de la perturbation. Toutefois, les phénomènes

de résonnances dans la boucle fermée environnement-senseurs-actionneurs compliquent l'édification d'une preuve de cette simple observation.

6 Conclusion

Ce travail a permis de dégager les limites de la méthode employée. Ainsi, cette méthode permet la construction de contrôleurs spécialisés capables de mouvements périodiques complexes. De plus, l'introduction d'une perturbation a abouti à la génération d'un contrôleur à fonctionnement double. Toutefois, l'évolution comme apprentissage n'est pas satisfaisante car elle conduit à des systèmes adaptés et non adaptatifs : les contrôleurs neuronaux sont figés et incapables de surmonter des situations inconnues (i.e. non prises en compte lors de l'évolution). Une piste susceptible de résoudre ce problème sera abordée dans le cadre du projet IRON et consiste à utiliser des synapses adaptatives [Floreano and Urzelai, 1999].

Références

[Baker, 1985] Baker, J. E. (1985). Adaptive selection methods for genetic algorithms. In Associates, L. E., editor, *Proceedings of an international conference on genetic algorithms*, Hillsdale.

[Baker, 1987] Baker, J. E. (1987). Reducing bias and ineffiency in the selection algorithm. In Associates, L. E., editor, *Proceedings of the second international conference on genetic algorithms and their applications*, Hillsdale.

[Ben Salem et al., 2002] Ben Salem, M., Monacelli, E., and Ben Amar, F. (2002). Performance study of an assistant robot. Soumis à la conférence ROMAN.

[De Jong and Sarma, 1993] De Jong, K. A. and Sarma, J. (1993). Generation gaps revisited. In Kaufmann, M., editor, *Foundations of genetic algorithms 2*, San Mateo, CA.

[Filliat, 1999] Filliat, D. (1999). Evolution de réseaux de neurones pour le contrôle d'un robot hexapode. Rapport de stage, LIP6.

[Floreano and Urzelai, 1999] Floreano, D. and Urzelai, J. (1999). Evolution of neural controllers with adaptive synapses and compact genetic encoding. In *5th European Conference on Artificial Life (ECAL'99)*. Springer-Verlag.

[Goldberg, 1989] Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley, Reading, MA.

[Guillot and Meyer, 2000] Guillot, A. and Meyer, J.-A. (2000). From SAB94 to SAB2000 : What's new, animat ? In Meyer, J.-A., editor, *Proceedings of the sixth international conference on simulation of adaptive behavior. From animals to animats 6*, Massachusetts Institute of Technology, Cambridge.

[Hénaff and Chocron, 2002] Hénaff, P. and Chocron, O. (2002). Adaptive learning control in evolutionnary design of mobile robots. In *IEEE-Internatinal Conférence on Systems Man and Cybernetics.*

[Hénaff and Delaplace, 1996] Hénaff, P. and Delaplace, S. (1996). Using back-propagation algorithm for neural adaptive control : experimental validation on an industrial mobile robot. In *ROMANSY'96.*

[Holland, 1975] Holland, J. H. (1975). *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, MI.

[Janikow and Michalewicz, 1991] Janikow, C. Z. and Michalewicz, Z. (1991). An experimental comparison of binary and floating point representations in genetic algorithms. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA. Morgan Kaufmann.

[Langaney, 1988] Langaney, A. (1988). *Les Hommes, passés, présent, conditionnel.*

[Meeden et al., 1993] Meeden, L. A., McGraw, G., and Blank, D. (1993). Emergence of control and planning in an autonomous vahicle. In Erlbaum, L., editor, *Proceedings of the Fifteenth Annual Meeting of the Cognitive Science Society*, Hillsdale, NJ.

[Pfeifer, 2000] Pfeifer, R. (2000). On the role of morphology and materials in adaptive behavior. In Meyer, J.-A., editor, *Proceedings of the sixth international conference on simulation of adaptive behavior. From animals to animats 6*, Massachusetts Institute of Technology, Cambridge.

[Rogers and Prügel-Bennett, 1999] Rogers, A. and Prügel-Bennett, A. (1999). Genetic drift in genetic algorithm selection schemes. In *IEEE-Transactions on Evolutionary Computation.*

[Ziemke, 2000] Ziemke, T. (2000). On "parts" and "wholes" of adaptative behavior : Functional modularity and diachronic structure in recurrent neural robot controllers. In Meyer, J.-A., editor, *Proceedings of the sixth international conference on simulation of adaptive behavior. From animals to animats 6*, Massachusetts Institute of Technology, Cambridge.